WordStar Backrub

Computing Now!

Canada's Personal Computing Magazine

February 1985

$3.25
MM70247

Home Computers

...for when they put the IBMs away.

- Commodore 16 Reviewed
- Commodore Plus 4 Reviewed
- Adam Star Patrol
- Colour Computer Print Utility
- Chirpee for the Commodore 64

- A Small Cat for CP/M
- IBM PC Utilities
- Apple Encryption
- iNet 2000

EXCELTRONIX

Special 1985 Wholesale Prices For You

We guarantee you the combination of BEST Prices & Service in Canada!

Star Micronics Gemini 10X 1 year warranty

• 120 c.p.s., • 816 characters print buffer, option 4K or 8K • standard parallel optional RS232C • tractor & friction feed. **$359.00**

Gemini 15X **$599.00**
Same as Gemini 10X. • 15.5" carriage

POWER TYPE LETTER QUALITY $639.00

• 18 CPS bidirectional logic seeking.
• 96 Petal Wheel

TTX Letter Quality

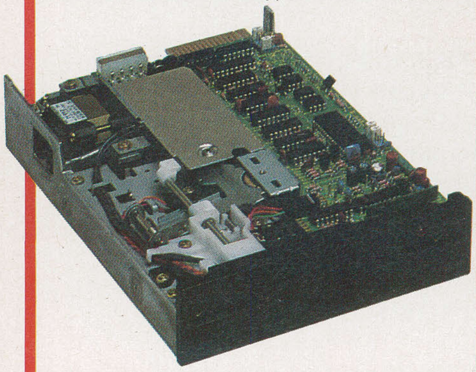
12 CPS Daisy Wheel	**$499.00**
EPSON FX 80	**$689.00**
EPSON FX 100	**$979.00**
EPSON RX 80	**$395.00**
Multiflex parallel Interface for Apple or IBM	**$ 69.00**

IBM COMPATIBLE KEYBOARDS

Cherry	**$149.00**
Maxiswitch	**$149.00**

IBM Compatible Disk Drive

SA455 (with warranty) **$195**



CASES FOR YOUR IBM Compatibles

• Hinged top cases allowing easy access by opening top half of the case **$74.95**
• Standard Sliding Cover Case **$69.00**

Hinged Case with 90W (max.) Power Supply with fan **$178.00**

As above with 175W (max.) Power Supply and fan **$228.00**

APPLE COMPATIBLE DISK DRIVES

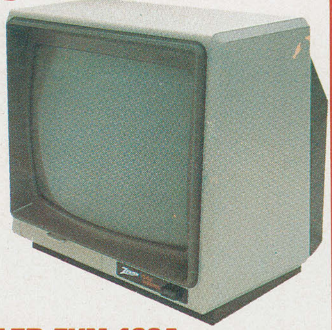
FAMOUS MULTIFLEX DISK DRIVE

$199.00
1 year full warranty.

Features:
• Apple compatible • Attractively packaged • Professionally built and tested • Canadian Made. We believe that Multiflex put out more drives in the last year than all other Canadian manufacturers combined.

Note: For $179 we offer 120 day warranty. However for 10% more get a full 1 year warranty.

Monitors Zenith Data Systems



BEST SELLER ZVM 122A • 12" diagonal screen • non-glare amber display • composite input • 25 lines x 40/80 characters **$139.00**

BEST SELLER ZVM 123A • 12" diagonal screen • non-glare green display • composite input • 25 lines x 40/80 characters **$129.00**

ZVM 133 • 13" diagonal screen • RGB input • 25 lines x 80 characters • 640 x 240 pixels green screen only switch • 16 colours including PC brown **$799.00**

APPLE MACKINTOSH $2549.00

Peripherals for your Apple:

Z80	**$ 49.00**
80 x 24 Video Card with Soft Switch	**$ 69.00**
16K RAM Card	**$ 49.00**
128K RAM Card	**$159.00**
Parallel Card with Cable	**$ 65.00**
Serial Card	**$ 69.00**
Apple Programmer	**$ 65.00**

AMDEK COLOUR 3+ **$389.00**
RGB Monitor 90 Day
RCB Card for Apple 2 +, or 2 E **$99.00**

300 BAUD MODEM for your APPLE $79.00

Auto Answer, Dial, Direct, Con. Plugs right into your Apple, comes with software.

1200 BAUD EMP MODEM $419.00

Hard Disk Drive

Seagate (industry favoured)
10 MEG. slimline **$895.00**
10 MEG Seagate, slimline drive and hard disk controller. This controller can handle up to two 10 MEG had drives.
Lowest Price of **$1,159.00**
Cable extra **$38.00**

MEMORIES, At prices you can't forget! All prime stock and we guarantee the best price.

4116 150 or 200ns	**$ 1.49**
4164 150ns	**$ 4.75**
Sets of 9	**$39.95**
41256K 150ns	**$27.95**
2114 200ns	**$ 2.50**
6116 150ns	**$ 6.95**
2716 450ns	**$ 4.99**
2716 300ns	**$ 5.99**
2732 350ns	**$ 6.99**
2764 300ns	**$ 9.95**
27128 300ns	**$19.95**

Special Parts for your IBM & Apple Compatibles

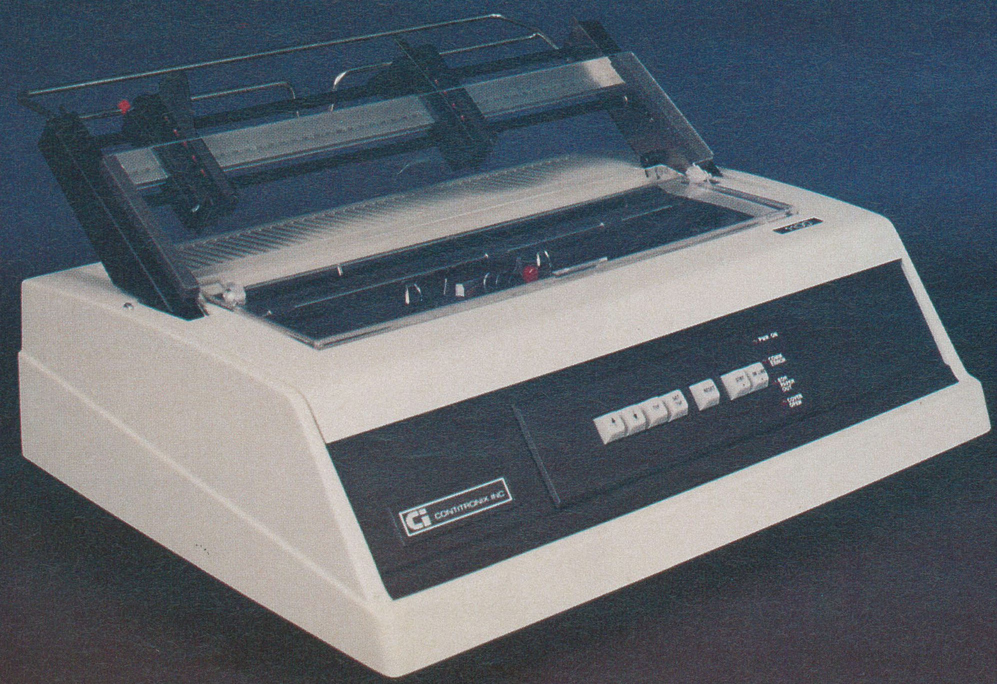
Prices good till March, 1985

8087	**$219.00**
8088M	**$17.50**
8237A-5	**$18.95**
8250	**$11.50**
8253A-5	**$ 8.65**
8255A-5	**$ 7.95**
8259A	**$ 7.95**
8284A	**$ 9.45**
8288	**$18.95**
NEC765/8272 Equiv.	**$18.95**
74CS322	**$ 6.95**
62 Pin Card Edge Connectors	**$ 2.19**
5 Pin Din Connector	**$ 1.29**
100ns Delay Line	**$ 8.95**
6502 CPU	**$ 5.99**
Z80A CPU	**$ 4.99**
68A45 CRT cont.	**$10.79**
TMS99532 FSK Modem	**$19.95**

All Parts in our new 24 page catalogue except Memories and other parts which are advertised on this page are 10% Off, for any studen bulk order discounts, can be arranged!

Circle No. 11 on Reader Service Card

Concorde Breaks the Price Barrier



A 45 cps Daisy Wheel Printer 1495^{00} *Suggested Retail*

The CI Concorde 1100 Printer comes by its name honestly. Its features soar above the competition, while its price stays at ground level.

Beneath a sleek exterior is tucked a simplified mechanism that powers one of the fastest printers around (45 cps). The Concorde 1100 is designed to handle all printing "whisper" quiet (60 dbA) on a generous 13.2-inch printline. It will tackle spreadsheets and graphics as easily as it will all correspondence.

And, first class extras come standard. The popular 96 character print wheel allows access to a wide variety of fonts and pitch sizes, while the print ribbon cartridge is clean and easy to change. Both are Diablo*-compatible and widely available. The Concorde's internal buffer is expandable to a full 40K so that lengthy print runs don't tie up the computer. Concorde also comes with friction platen and serial or parallel interface. For specialized uses, options include cut-sheet feeder and paper tractor.

For performance without the high price tag, fly with the Concorde.

TORONTO: 55 Torbay Road, Unit #2
Markham, Ontario L3R 1G7
(416) 477-0828

HEAD OFFICE: 17507-107 Avenue
Edmonton, Alberta T5S 1E5
(403) 489-8400 Telex: 037-42881

Distributed exclusively in Canada by

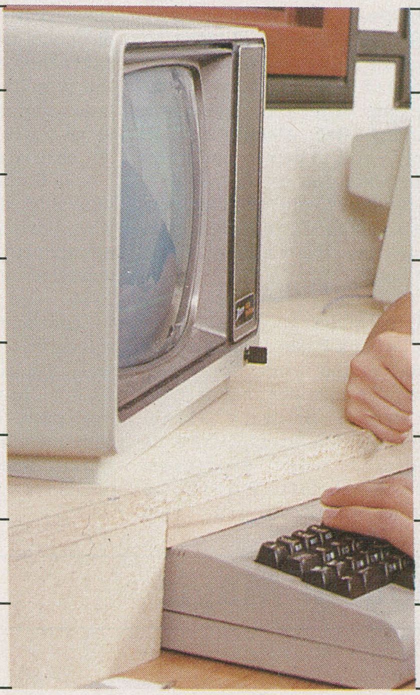
Computron

*Diablo is a registered trademark of Xerox.

Circle No. 30 on Reader Service Card

Computing Now!
Canada's Personal Computing Magazine

Vol. 2 No. 11
February 1985



Sectors

Published by
Moorshead Publications
Suite 601, 25 Overlea Blvd.
Toronto, Ont. M4H 1B1
(416) 423-3262

EDITOR
Steve Rimmer

ASSISTANT EDITOR
John Rudzinski

DIRECTOR OF PRODUCTION
Erik Blomkwist

CREATIVE MANAGER
Ann Rodrigues

PRODUCTION
Douglas Goddard
Naznin Sunderji

CIRCULATION MANAGER
Lisa Salvatori

ADVERTISING
Rick May
Montreal (514) 735-5191
Vancouver (604) 688-5914
Western United States
The Brassett Co., Inc.
(213) 802-2571

Publisher: H.W. Moorshead; **Executive Vice-President:** V.K. Marskell; **General Manager:** S. Harrison; **Controller:** B. Shankman; **Accounts:** P. Dunphy; **Reader Services:** C. Wyatt, M. Greenan, H. Brooks, R. Cree; **Advertising Services:** D. Whitehead

NEWSSTAND DISTRIBUTION
Master Media, Oakville, Ont.

SUBSCRIPTIONS
$22.95 (12 issues) $37.95 (24 issues)
Published 12 times a year

For US rates add $3 per year.
Other countries add $5 a year.

Moorshead Publications also publish Electronics Today, Computers in Education, and Software Now!

POSTAL INFORMATION
Second Class Mail Registration No. 5946. Mailing address for subscription orders, undeliverable copies and change of address notice is:
Computing Now!, Suite 601, 25 Overlea Blvd., Toronto, Ontario, M4H 1B1
ABC
Printed in Canada ISSN 0823-6437.

COMPUTER PRESS

Obituary

DECEASED – The **Coleco Adam**; of abandonment, in West Hartford, Connecticut.

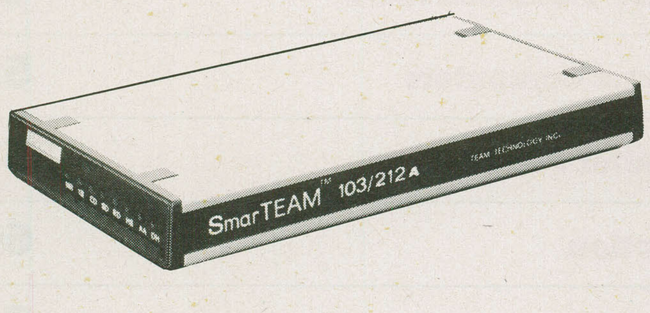
In a post-Christmas move perhaps not surprising to industry analysts and critics of the computer system, *Coleco Industries Incorporated* discontinued manufacturing the **Adam** system. By selling existing Adam inventories to an un-named American retailer, the company has abandoned the highly competitive computer marketplace for good, though it will continue to sell the "marginally profitable" Colecovision game system and the Cabbage Patch Kids line.

Originally released here in late 1983, the Adam shook the industry by offering a letter-quality printer as part of the system for a purchase price equalling that of many daisywheel printers alone. Together with a fast tape drive, an integrated word processor, mock-Applesoft BASIC, a detachable keyboard, two joysticks and a game, the Adam made for a fairly attractive package when it was introduced.

Some complaints did arise from some purchasers of the unit, however. SmartBASIC tended to erase itself every now and then, as did many tapes placed on or near either the system or the printer. As the printer contains the system's power supply, it must be attached at all times. Interfacing a faster, dot-matrix printer is impractical.

In Canada, "no panic buttons were pushed" by Coleco Canada when hearing of the American announcement. The Adam will be marketed here in 1985 as it was in 1984, and the company will "...continue to expand the Adam's software base" throughout the year, according to spokesperson Joanne Moreau of the Montreal-based company. The company maintains that Adam service and customer support is being continued.

Modem Magic



MISSISSAUGA, ONTARIO – The exclusive Canadian rights for the **Smarteam 103/212A Modem** have been acquired by *Budgetron Incorporated*.

The Smarteam 103/212A is a fully Hayes compatible standalone modem, with switchable 0-300/1200 baud auto-answer, auto-dial, auto-speed selection and full- or half-duplex operation capabilities. It can operate under a variety of available software, including PC Talk, PC Crosstalk and SmartCom II, as well as its own *TEAMTALK* for the PC, or *MacTEAM* for the Apple Macintosh.

The unit includes a two year warranty, and is presently being offered with *TEAMTALK* and an RS-232C serial port interface cable as a bonus. Suggested list price for the modem is $495.00.

Budgetron Incorporated is at 1601 Matheson Boulevard, Unit 3, Mississauga, Ontario L4W 1H9 (416) 624-7323.

Show Me

TORONTO, ONTARIO – The third annual conference and exhibition presented by *Communications Kraft, Incorporated* and endorsed by the *Data Processing Management Association*, **Software Panorama 1985** will be at Toronto's Royal York Hotel from May 22nd to the 24th, 1985. The show features the first international conference of **Computers and Human Development**, sponsored by the *Software Developers' Association.*

Software Panorama 1985 celebrity guest speakers include former U.S. Secretary of State General Alexander Haig, Astronaut-businessman James Lovell and Thomas J. Peters, author of *In Search of Excellence.* Seminar leaders will include Professor Warren McFarlan or Harvard and Professor Richard Byrne of UCLA.

Displays at the panorama will include those of software packages, computer and data processing equipment, graphics, data communication, software consultants, learning institutions, related supply and service vendors, software demonstrations and service bureaus. Projected attendance of the show has been set at over 4,500 professional data processing people, software personnel, multi-level management personnel from business, industry and government sectors.

Interested parties may contact Communications Kraft Incorporated through Andy Berneshawi, Director of Marketing, 200 Consumers Road, Suite 200, Willowdale, Ontario M2J 4R4 or call (416) 494-3416

On the Air

REDMOND, WASHINGTON – A first in North American broadcasting occured recently when Washington-based radio station KAMT-AM broadcast a picture of U.S. President Reagan. The picture was sent using a technique called *Softcasting* – software broadcasting. The developers of Softcasting chose this method (a MacPaint drawing transfer) to demonstrate how data and computer-generated pictures can be transmitted by radio.

Next Month In Computing Now!

Graphics

In the next edition of Computing Now! we'll be having a dig into that most interesting of microcomputer applications, graphics. Quite a number of recent systems have emerged with incredibly powerful graphics facilities . . . we'll be looking at some of the techniques involved in using them.

We'll also be having a peer at some of the graphics toys one can hang on a micro. For example, there's a pretty powerful new box from Polaroid which allows one to produce instant prints or slides from a micro's video display without all the machinations of a camera and a darkened room. We'll be checking it out next month.

The Art of the Overlay

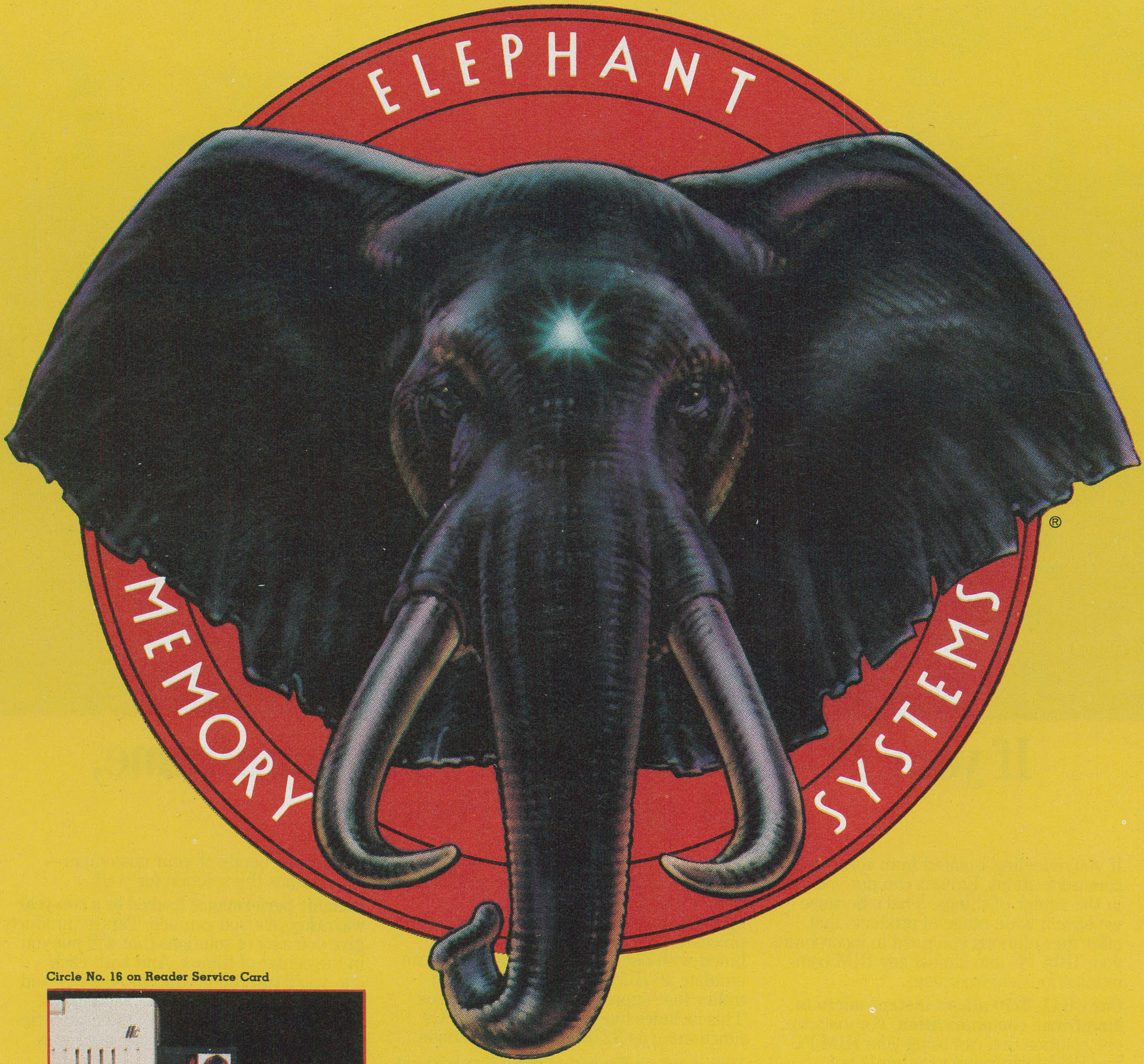
Writing an overlay for one of the more sophisticated derivations of MODEM7 can be a quick hack and a promise . . . or it can be a splendid, graceful interface of the programmer and his environment. Ya, that does sound like an ad for Ballet News, I suppose. In any case, in the next issue of Computing Now! we'll be checking out the techniques involved in writing a really powerful MDM overlay section.

The Tandy 1200

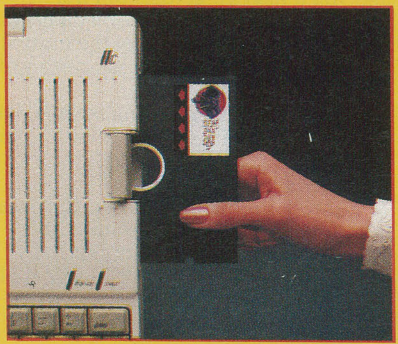
Definitely not what you want to hear, the latest computer to emerge from Radio Shack is all but perfectly silent. It has an integral hard disk to eliminate the gronching of a floppy, a pretty tiny speaker to nullify most music programs . . . even the keyboard is relatively quite. It features MS-DOS version two and is among the most IBM compatible boxes on the planet. We'll be checking under its hood shortly.

These features are in an advanced state of preparation. However, in endeavouring to keep Computing Now! as up to the minute as possible we reserve the right to change the contents of this issue prior to going to press.

TEST DRIVE AN ELEPHANT.™



Circle No. 16 on Reader Service Card



Elephant Floppy Disks are the perfect vehicle for storing and protecting data. Because Elephant never forgets. You'll get high performance that's 100% guaranteed for a lifetime of heavy use. So take them for a test drive. They're available now at your local computer showroom. And there's no waiting for delivery.

Dennison

ELEPHANT™ NEVER FORGETS.

For information call toll-free
1-800-387-5037
BC 112-800-387-5037 **Toronto (416) 495-9353**

Dennison Manufacturing Canada Inc.
200 Base Line Road East
Bowmanville, Ontario L1C 1A2

Commodore Plus 4 Review

Perhaps one of the most unusual things ever to be wrapped around a Commodore circuit board, the Plus 4 is a powerful computer with a host of integrated applications buried in there.

by Steve Rimmer

As it happened, when I asked for a Commodore 16 to review I got it and a little plastic friend to go with it, the Plus 4. At the time I'd never even heard of the Plus 4, and, for a while there was a lot of speculation that it was actually some sort of of Atari video game that had been heaved in the box by mistake. As it turned out, however, not only was it an authentic Commodore computer, but it had some really slick things happening within its art deco shell.

There are decided similarities between the two systems. For one thing, they both use what appears to be an identical implementation of BASIC. They support the same graphics and sound facilities.

The Plus 4, however, has heaps more memory and, more to the point, tricky integrated software that lives right there in the computer.

Three Score and Four

The Plus 4 boots up like pretty well any other Commodore computer, thinking for a moment and then blanking the screen and dragging BASIC out of bed. There's the traditional boot message at the top of the screen and... praise the heavens above or below, as the case may be... just over sixty K of RAM free. This is, of course, sixty real K's, that is, sixty kilobytes that you can actually use.

The Commodore 64, by comparison, has about thirty-eight kilobytes available for use under BASIC, the remainder having

been lopped off by BASIC itself.

These rippling fields of memory are a decent trip for programming the Plus 4. The system's BASIC, promising but badly restricted under the 16, is free to do all the stuff the hardware will manage, like high resolution graphics, and still leave enough RAM to write reasonable size programs to control the whole dog and pony show.

It may seem strange, then, to observe that the true power of the Plus 4 is not so much in its BASIC facilities but, rather, in the other software that lives in its ROMs. When you first boot the computer it programs its seventh function key with a SYS instruction that exits BASIC and launches into a massive integrated software package which is also resident.

By simply rapping the key the whole machine can be turned schizophrenic, and be transmuted into...

It becomes a word processor, actually.

The integrated software built into the Plus 4 is comprised of a word processor, a spreadsheet and a data base manager. The word processor more or less leads the list, as one calls the other two applications as one needs them from it.

Four and Twenty

There has been quite a host of word processing packages for the small Commodore computers which preceeded the Plus 4, but few of them work as well or as simply as the one which comes built into this box. It's not anywhere near as powerful as a serious word processing package, but it compares very favourably to anything else one could run on the Plus 4 without additional hardware.

The most pronounced limitation of any simple word processor for a home system is that it must contend with a small screen... all reality is forty characters across. Most people don't think in forty columns, and letters and manuscripts printed out in this form look pretty dreadful.

Actually, they look like they've been typed on a home computer.

The Plus 4's word processor overcomes this by doing a lot of lateral scrolling. It thinks it has a seventy seven column virtual screen... all the lines it creates are done so with this in mind. As you type it attempts to move its forty column window over this buffer as best it can to make your image of what you're typing fairly clear.

This approach works a lot better than it sounds like it should. What's more, the illusion of the window's scrolling sideways is handled very quickly, so there is no tedious waiting to deal with every time the system wants to print the forty-first character in a line.

Specs

System:	**Commodore Plus/4**
Application:	**Home computer**
Operating System:	**Commodore BASIC**
Memory:	**64K; 60K from BASIC**
Mass Storage:	**1531 datasette, 1541 or 1551 disk drives (all optional)**
Processor:	**8501**
Software Included:	**BASIC, word processor, spreadsheet, filer**
Software Available:	**Not much yet**
Manufacturer:	**Commodore**
Price:	**$529.95**

As you type, the screen scrolls over. If you go streaking past the seventy-seventh character, or hit a carriage return somewhere along the line, it will pop back to the left side of the virtual screen.

One of the decided limitations of the word processor is that it shows you the text as it is entered, as opposed to what it will look like when it's printed. Thus, for example, if you are half way in the middle of a word when you run out of space on a line it will simply split the word and wrap around to the next line.

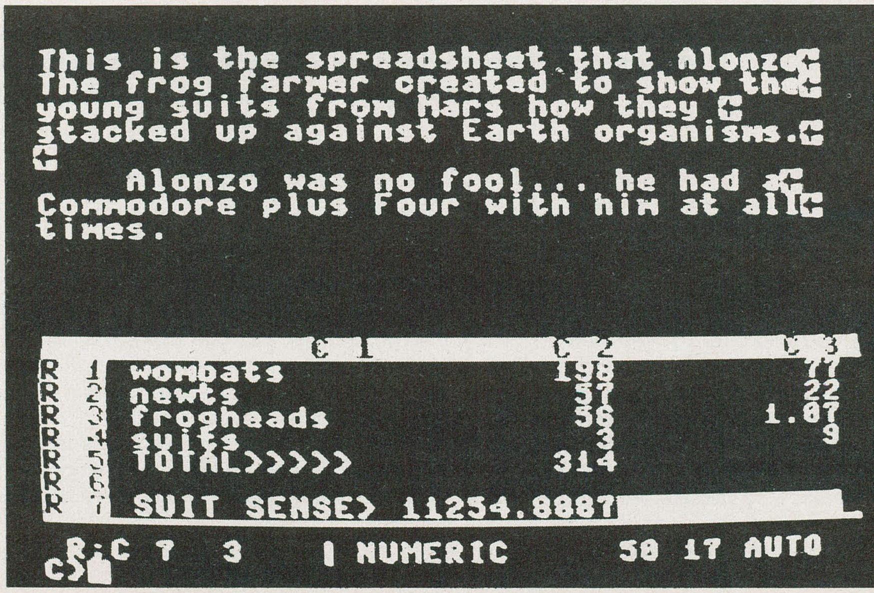
The program formats the text properly, with properly ended lines, when you go to print it out. However, you do need some imagination to see it this way when you're looking at it on the screen, especially if you try to do tricky formatting.

Having typed something into the Plus 4, it can be manipulated with a fairly decent array of the sorts of things word processors usually do. One can, for example, cursor around the screen inserting and deleting characters to one's heart's content. These facilities behave much like they do under BASIC.

Actually, it looks like the designers of the word processor scooped quite deeply into the BASIC ROMs. For example, the double quote key produces a single quote symbol, as double quotes are very significant to BASIC's editor.

There are a few peculiarities in the editor. In entering text into the word processor, hitting a carriage return causes an inverse arrow to be displayed on the screen at the end of the line. If you cursor up to a point in existing text and try to insert a carriage return... that is, to create a new paragraph... the carriage return will show up but the part of the line to the right of it will vanish into the twilight zone, never again to grace the multi-coloured phosphor.

Under the Plus 4's editing facilities, inserting say a half a dozen characters into a line entails hitting the insert key a half a dozen times to open up some space for them to live in. This is a bit tedious if you fancy plugging in a new paragraph or two, so one can scoot up and insert a whole blank line.



A split window word processing document and spreadsheet

COMPUTER MAIL ORDER

CANADA

TELEX 06-218960 TELEX 06-218960

ONTARIO/QUEBEC	TORONTO	OTHER PROVINCES
1-800-268-3974	828-0866	1-800-268-4559

We Carry a Wide Selection of Software From:
Artworx, Broderbund, Commercial Data, Creative Software, Epyx, Atarisoft, Hes, Sirus, Microspec, Synapse, Thorn, Spinnaker, Timeworks + Many More. Call us for prices!

(All equipment under full warranty in Canada.)
Price subject to change without notice.
(All orders Shipped within 48 hrs, upon payment.)

COMPUTERS

- APPLE .... CALL
- COLUMBIA .... CALL
- COMPAQ .... CALL
- EAGLE .... CALL
- FRANKLIN .... CALL
- HEWLETT PACKARD .... CALL
- IBM .... CALL
- SANYO .... CALL

HAND HELD

HEWLETT PACKARD
- HP11C 94.99
- HP12C 149.99
- HP15C 149.99
- HP16C 149.99
- 41CV 279.00
- 41CX 399.00

DISK DRIVES

INDUS
- GT ATARI Drive 429.00

MSD
- MSD-1 C64 Drive 529.00
- MSD-2 C64 Drive 849.00

PERCOM .... CALL

RANA
- 1000 (Atari) 449.00
- Elite 1 .... 389.00
- Elite 2 .... 579.00
- Elite 3 .... 799.00

TARGA .... CALL

TALLGRASS .... CALL

TANDON
- TM-100-2 DS/DD 279.00
- TM-50-2 DS/DD 349.00

MODEMS

ANCHOR
- Mark VII (AA/AD) 193.00
- Mark XII (1200 Bd.) 489.00

HAYES
- Smartmodem 300 .... 339.00
- Smartmodem 1200 .... 799.00
- Smartmodem 1200B 645.00
- Micromodem II e 427.00

NOVATION
- J-Cat 149.00
- Smartcat 103 .... 289.00
- Smartcat 103/212 .... 649.00
- Autocat 356.00
- 212 Apple Cat 929.00
- PC-Cat 549.00

MONITORS

AMDEK
- Video 300 .... $229.00
- Video 300 A 239.00
- Video 310A IBM 249.00
- Colour 1 + ... 429.00
- Others .... CALL

BMC
- 12" Green 125.00
- 12" Amber 135.00

GORILLA
- 12" Green 129.00
- 12" Amber 139.00

NEC .... CALL

PANASONIC .... CALL

PRINCETON GRAPHICS
- HX-12 .... 749.00

QUADRAM
- Quadcolor RGB 809.00

SAKATA .... CALL

TAXAN .... CALL

USI .... CALL

ZENITH
- Amber 159.00
- Green 149.00
- Colour 795.00

CARDS

AST RESEARCH .... CALL

QUADRAM .... CALL

HERCULES
- Graphic card 525.00

PLANTRONICS
- Colorplus 699.00

ACCESSORIES

KEYTRONIC
- KEYBOARDS .... CALL

KOALA PAD
- All Computers .... CALL

INTERFACES

Batteries Included
- Bus Card .... CALL
- 80 Col. Card .... CALL

CARDCO
- Int/Graphics 99.00

TYMAC
- C64 to EPSON OKI 148.00

MPC
- Apple Dumpling GX. .... CALL

PKASO
- Universal Card 225.00

PRINTERS

BMC
- 401 Letter Quality $923.00
- BX-80 Dot Matrix 399.00

C. ITOH
- Gorilla Banana 299.00
- Prowriter 8510P 550.00
- **Prowriter 1550P** 899.00
- F10-40 .... 1660.00
- F10-55 .... 2175.00

DAISYWRITER
- 2000 .... 1499.00

COMRITER
- CW2. .LQ 12CPS 749.00

EPSON
- RX80 .... 399.00
- RX80FT 525.00
- FX80 .... 615.00
- FX100 .... 975.00

JUKI
- 6100 .... 695.00

OKIDATA
- 82 .... 499.00
- 83 .... 875.00
- 84 Par 1200.00
- 84 Ser 1379.00
- 92 .... 649.00
- 93 .... 1059.00

MANNESMANN TALLY
- Spirit 80 .... 399.00
- 160L 895.00
- 180L 1229.00

SMITH CORONA
- L1000 .... 589.00

STAR
- Gemini 10X 399.00
- Gemini 15X .... CALL

BROTHER
- HR 15 .... 689.00

TTX .... CALL

DISKETTES

BASF
- BA1D SS/DD W/Case 28.99
- BA2D DS/DD W/Case 35.99

ELEPHANT
- EL1 SS/DD 27.99
- EL2 SS/DD 29.99
- EL3 DS/DD 39.99

MAXELL
- MXD1 SS/DD 37.99
- MXD2 DS/DD 54.99

NASHUA
- ND1D SS/DD W/Case 26.99
- ND2D DS/DD W/Case 38.99

VERBATIM
- VE1 SS/DD 31.99
- VE2 DS/DD 43.99
- VE1C SS/DD W/Case 34.99
- VE2C DS/DD W/Case 46.99

Dealer Enquiries Invited
Write or telephone for our FREE catalog

To order: Send money order, certified cheque, personal cheques must clear our bank, VISA or MASTERCARD. (Include card # and expiry date & signature) Add 5% for shipping and handling. Minimum $5.00 per order.

Ontario residents add 7% P.S.T.

Computer Mail Order Canada 2505 Dunwin Drive, Unit 3 • Mississauga, Ontario L5L 1T1

Circle No. 8 on Reader Service Card

Commodore Plus 4 Review

Having inserted a blank line, you can type in text. However, there's a catch here too. If you type too much text into the line it will wrap off the end and overwrite the next line. You kind of have to watch the screen quite carefully when you're doing this.

The word processor has a fair assortment of text formatting instructions... they're a bit obtuse at first, but you can get used to them. There are also search and replace and block manipulations available, plus a command to link multiple short files together.

The word processor does have a finite file length, being memory based.

While I wouldn't want to try writing a book on it, the little word processor in the Plus 4 is a decent bit of software for what it does. It feels fairly friendly, and is decently human engineered... two aspects which are rather more important than a book full of obscure features.

Spread Them Sheets

The spreadsheet package in the Plus 4 is integrated with the word processor from a number of points of view. To begin with, it is called from the word processor's command line and, in fact, one can bounce back and forth between the two applications at will with no loss of data.

There is also a fairly reasonable degree of consistency between the command structure of the two packages. Finally, there is a primitive windowing facility built into the system which allows one to view half the word processing screen and half the spreadsheet simultaneously.

Viewed in isolation, the spreadsheet program of the Plus 4 is of only moderate abilities. It supports fifty rows of seventeen columns, which makes for a fairly tiny sheet. As is the case with the word processor, one scrolls over this sheet with the screen window.

The scrolling and cursor movements of the spreadsheet are a bit peculiar, in that the cursor up and down arrows do what you'd expect them to, but one uses two of the system's function keys to move laterally. The cursor tight and left keys move the cursor on the command line.

Moving about on the spreadsheet and entering things into its cells is fairly straight up... once you get around that little detail with the cursor keys. There are a couple of unusual bits to get used to. As a f'rinstance, if you cursor over to a cell which contains a formula, the status part of the command display will insist on showing you the result of the formula, that is, the cell's contents as displayed on the sheet, rather than the formula itself unless you specifically instruct it not to each time you encounter a formula cell.

There are also one or two truly annoying aspects to the sheet. If you insert a blank row or column in the sheet everything will over down or over, as the case may be, but the sheet will not adjust the formulae. As such, one must pop through the whole sheet locating the formulae and fudge them all by hand. This is something of a downer.

On the other hand, the spreadsheet has a pretty decent repertoire of advanced bits, such as replicating chunks of a sheet and a user selectable recalculation mode. Recalculations are none too swift, but they can't get intolerably long because you can't create particularly big sheets.

The spreadsheet allows for handing data back and forth between itself and the word processor. It also has a fairly basic graphics display feature, which allows for doing graphs based on spreadsheet data made out of hash mark characters.

Other Bytes

The data base manager which comprises the third part of the integrated package doesn't really require a lot of comment. It can do the things one expects of a small database manager. The record editing facilities are a bit better than most. The facilities for doing complex sorting are a bit worse.

The database can be buckled onto the word processor to allow the system to produce form letters... what the suits call mail merging. This is actually a pretty sophisticated capability for a package of this class.

The software which comes with the Plus 4 is, on the whole, fairly well written. It shouldn't be mistaken for serious business software... it isn't... but it will handle the sorts of casual tasks computers are used for at home quite nicely. If you are up for the sorts of things the smiling people on the Commodore commercials seem to spend their entire lives doing you'll be able to handle them on the Plus 4.

The Plus 4 itself... the computer, as opposed to the denizens of its ROMs... is a fairly peculiar little beast. For one thing, it doesn't look at all like a Commodore. Its keyboard is something of a treetoad's backscratcher... not really gross, but several notches below the ones they put in the earlier Commodore machines. The function keys are tiny plastic slivers up above the normal keys, while the cursor movers are trendy plastic arrows that just plain act weird.

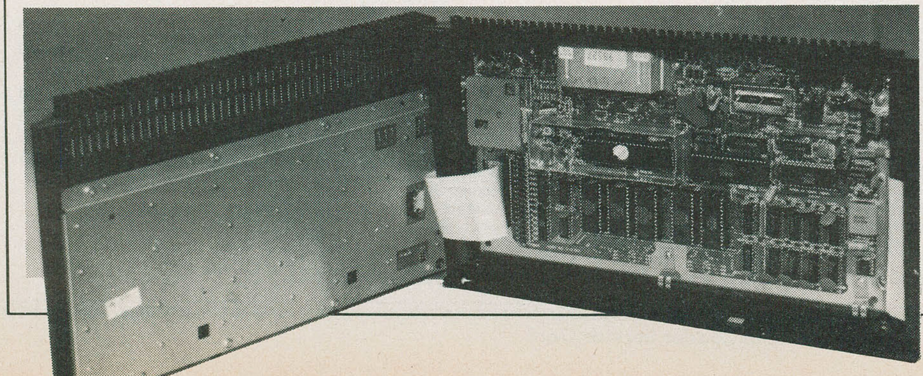
The case of the one I got to look at developed the annoying habit of going for strolls across the table while I was typing. It's extremely light.

The Plus 4's hardware seems to be pretty solid stuff. It endured several days of continuous power without smoking the great banana and returning to its maker. It also never glitched out in that time, something computers tend to like to do around here in the crackling dry air of mid–winter.

Like the 16, the Plus 4 supports a 1541 disk drive, joysticks and a user port, which can, in turn, drive a modem and any custom peripherals you care to develop for it. The peripheral facilities will also be friendly with a printer... so long as it's a special Commodore printer. There's an on–board television modulator, and you can get the composite video out of the machine through a special cable to drive a colour monitor. As it has been since the beginning of time, poor users can plug a datasette into the Plus 4, although this datasette uses a different connector than have any of its predecessors. Make sure you buy the right box.

None of the integrated software seems to be the slightest bit interested in using the datasette as a mass storage device. You'd pretty well have to have a disk drive to make practical use of this software.

The Plus 4 is a pretty fine little box for what it costs. It's a powerful home computer... a good choice if you want to get into all this technology without shooting the moon all at once. Many users will outgrow its modest capabilities in time, but this is probably true of most computers at the moment. Even the truly massive systems run out of steam sooner or later. **CN!**



Commodore 16 Review

Always up for a good afternoon with the corner numerologist, the designers at Commodore have come up with another two digit plastic box. This one features a powerful BASIC, better graphics and even more money for advertising.

by Steve Rimmer

Obsolete Commodore computers can pose no small end of problems. If you're an avid follower of Bahamian high technology, and are up for acquiring all the newest toys as they show up at Canadian Tire, you'll probably eventually find yourself with something of a disposal hassle.

We had a Commodore 2001 PET kicking around the battlements for an unusually long time. This was a particularly nasty case, having a gargantuan cast iron box topped by a massive black and white monitor, all of which precluded its simply being shelved somewhere. The shelves were always either too small or, most often, too weak.

We eventually gave it to my father. The last time we were over at his place he was using it to prop up all the manuals which went with it, an application I confess we never thought of. I think he's investigating the possibility of using the monitor part with an Apple clone.

The newer Commodore machines are, to be sure, designed with their eventual disposal considerably further to the fore. To begin with, they are all moulded from the same dies, which means that if you come up with a use for an abandoned VIC 20 it'll be just as applicable to a 64 when something new shows up to replace it.

The case design which has been finding itself around the smaller Commodore systems is a lot more functional than the older style metal PET boxes too. It doesn't have a built in tube, which means that it can be forgotten under beds, behind books and in other small and otherwise unused niches and alcoves. Its wedge shape makes it mildly applicable to such things as stopping very sloppy doors, shoveling snow and playing fetch with a large dog.

All this borne in mind, the new Commodore 16 is a familiar and pleasing sight. It looks like exactly like a VIC or a 64, except that it's jet black. However, as we shall see, it's considerably more than just a cheaper 64. The 16 embodies what amounts to the first improvement in PET BASIC since the language was first spilled into the 2001 ROMs almost a decade ago.

RUN

The Commodore computers which have turned up for use by beginners have had a really weird assortment of features. The 64, for example, had an uncool primitive BASIC... at least by contemporary standards... and a lot of pretty decent facilities which could only be accessed through an

It also had a rather unusual assortment of hardware in reflection. For example, it supported sprites but no BASIC supported high resolution graphics... at least, not without some additional plastic action. It did have sixty four K of memory, but this was largely something for the boys in advertising, as half of it was unusable in most applications.

The Commodore 16 is a much better trip in quite a number of respects. It has much of the hardware power of the 64, a much more sophisticated language to play with, reasonable compatibility with earlier Commodores, as these things go, and a better smattering of bells and whistles. It's also a bit cheaper.

The first thing you'll probably notice about the 16 when you boot it up is its allotment of RAM. As with all Commodore systems, the computer steals part of its memory for its own internal housekeeping. You actually get a little over twelve K to play with. This is pretty good for programming... you can't write a program to sort the sum of man's knowledge in this space, but most reasonable code can be dealt with in ten K or less.

The only hassle with all this is the high resolution graphics. If you're already somewhat into computers, and have played with a few systems before, you'll probably want to try out the 16's graphics facilities early on. This has a very profound impact on the machine's memory.

The graphics mode immediately snatches a ten K block of memory for itself, leaving only a couple of K for a program to drive the graphics.

I would imagine that memory expansion packages should be a popular bit of silicon for 16 owners.

This minor detail aside, the graphics for the 16 are extremely good. They are, to begin with, fully supported from BASIC. What's more, the BASIC commands are well thought out and extremely rich. There is an assortment of graphics modes of varying resolution and colour facility; DRAW to plot points, lines and shapes, LOCATE to move the graphics cursor and PAINT to fill things in. In addition, there is a facility for copying graphics data from the screen into BASIC arrays and then putting it back elsewhere on the tube.

There are also BOX and CIRCLE commands to draw specific shapes. The latter is extremely flexible, allowing for elipses and arcs as well.

The graphics facilities of the 16 from BASIC are extremely well thought out and easy to use. They are, however, somewhat slow, making things like animation for

Commodore 16 Review

games a bit difficult to get together.

If you aren't up for all this bit mapped splendor, the 16 still supports the same set of PET graphics characters that all its predecessors have. This has a number of applications. To begin with, these things can be laid down on the phosphor much more rapidly than can high resolution images. You can do limited real time animation with block graphics from BASIC and have the whole affair look reasonably convincing.

The block graphics characters have been central to programming Commodore systems since the dark ages and, as such, much of the tottering library of programs that has evolved for PETs and their offspring can be easily adapted for the 16.

Along with all the visual things the 16 can manage, it has sound facilities as well. All civilized computers are up for this to some extent, but many do it through heavy machine language or the aforementioned PEEKing and POKEing.

There are two sonic statements in the 16's BASIC. The VOL command sets the output level at one of eight graduations, while SOUND allows one to control the warblings of one of two voices plus white noise... this latter bit being for cosmic detonations, impacting stellar warp fighters and other cataclysms.

The sound that the 16 spews forth... it'll come from the speaker of your TV if you're using one... resembles that of an oboe in the hands of a master player right after the instrument has been flattened by a truck. It's about par for computer generated music, I suppose. It has but a single timbre... the always popular thick square wave noise that most systems produce... and only moderate intonation. The manual offers a table of values which correspond roughly to musical notes. If you stick to a key of C it isn't too offensive.

Back To BASIC

The variations on BASIC which have accompanied Commodore hardware have always been particularly decent examples of human engineering, and that of the 16 is no exception. For example, one can still edit lines by simply cursoring up to them on the screen, changing the offending bits and hitting return. However, there are a number of enhancements to this.

The graphics facilities of the 16 from BASIC are extremely well thought out and easy to use.

The first fairly neat bit in the 16 is the existence of HELP. If a program running on the 16 encounters an error and barfs, typing HELP will list the line with the uncool portion flashing.

In using HELP one finds that hitting the f8 function key will also invoke HELP. There's an explanation for this... the function keys are programmable. You can load each with whatever string you want, all ready to spew out onto the tube when you rap one on the head. Thus, one could put useful strings, like RUN, or LIST, into the keys to avoid having to type them.

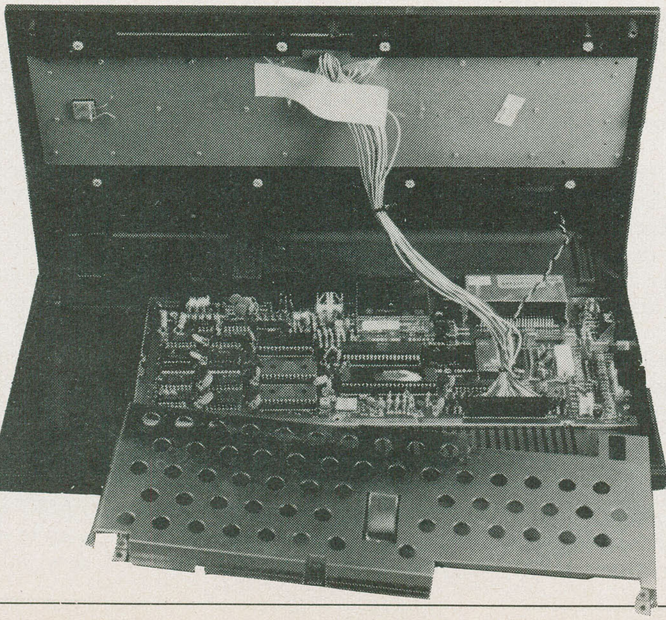
In fact, the function keys are loaded with an assortment of useful strings when the computer powers up. The internal verbs are

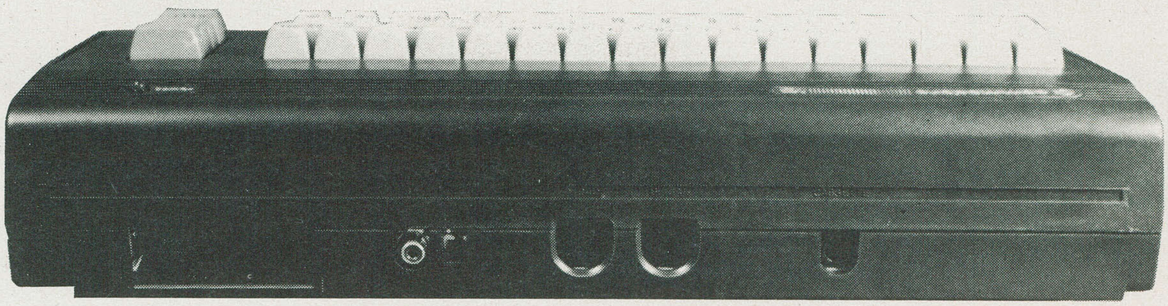
1. GRAPHICS
2. DLOAD '
3. DIRECTORY
4. Screen Clear
5. DSAVE '
6. RUN
7. LIST
8. HELP

You can change them from BASIC if you feel so moved.

A lot of what the 16's BASIC offers over earlier versions isn't particularly flashy, although it does make for writing tighter programs with less effort. There is, for example, AUTO and RENUMBER. The system's disk facilities are now supported by BASIC keywords dedicated to the task. These commands are a bit unwieldy as compared to the disk file facilities of other phila of BASIC, but they are a decided improvement over having to treat the disk drive as a purely logical IEEE device, the lot of earlier Commodores.

Other hidden wombats include a PRINT USING statement to allow for slick formatted displays. There is also a new control structure in BASIC, the DO loop. In some ways analogous to a FOR NEXT loop, this thing allows for looping on the condition of a value, rather than for a specified number of iterations. It's something which is





found in most higher level languages, but rarely in BASIC. As you get into writing code this thing will become extremely useful.

Finally, there is a very handy MONITOR command. Hit this thing and the system will dump you into its own built in machine language monitor. The 8501 processor which drives the 16 is essentially the same as the older 6502, so programming it at the machine language level is fairly straight up.

The monitor which lives in the 16, TEDMON, is about everything a monitor should be. It will assemble and disassemble code, hunt for strings, load and save machine language programs or blocks of bytes, fill and edit memory, move memory, compare memory and allow one to check out the condition of the registers. It lacks single stepping and tracing, but these features seem to be rarely found in small monitors.

There isn't much one can say about monitors... they're pretty top down and, assuming they work properly... this one does... they're about as interesting as doorknobs. However, the availability of one on board for the 16 is pleasing. It will prove to be a worthwhile tool for anyone who gets into programming the little black troll.

And We Bid You...

The 16 is a pretty decent computer for what it costs and what it's designed to do. Memory is cheap... I think the designers of this machine have been a bit tight with it. However, most users won't feel the pinch for a while, at least until they get into more serious programming.

The 16 supports a fair assortment of peripherals, some of which are the same as those for the 64 and the VIC. The 1541 disk drives, for example, are comfortable with the 16 as well. It will drive a colour TV or a composite monitor.

The 16 that we got to look at came with one of the least useful books I've ever encountered in the same box as anything more complicated than a blender. Not only is it badly printed and a bit gross in its colour scheme... it also says almost nothing about using the computer. The BASIC, for example, isn't even touched on.

Specs

System:	**Commodore 16**
Application:	**Home computer**
Operating system:	**Commodore BASIC**
Memory:	**12K useable**
Mass Storage:	**Datasette or 5 1/4" disk (both optional)**
Processor:	**8501**
Software Included:	**BASIC**
Software Available:	**Not much yet**
Manufacturer:	**Commodore**
Price:	**$199.95**

It's also probably worth pointing out that little of the existing commercial software for the VIC and the 64 will work with the 16. You may have to wait a while before you can kill any particularly decent aliens on the thing.

I used the manual for the Commodore +4, which embodies the same BASIC, to program the 16. I'm sure there *is* a proper BASIC manual for the 16... if you're thinking about giving table space to one, make sure you score a book.

The Commodore 16 is a good little box to consider if you're up for getting into computers. As beginner's machines go it's inexpensive, moderately flexible and easy to use. It has a lot of features which are enormously fun to play with, and a BASIC which is right up there with the best examples of computer languages... at least, as far as BASIC goes.

As with all of the small Commodore computers, I can't see the 16 being of much use for serious stuff... writing letters, doing mailing lists and so on... but it seems somehow inhuman to lay such mundane stuff on a box that has such potential for creating so much virtual confetti.

Argh, Billy, if ye can think of somethin' that rhymes wi' thirty two and a half we can write us a song, make a computer te be goin' with it and become rich men, me bucko... **CN!**

The bubbling, seething morass of on line information systems which has oozed out of the data processing industry has succeeded in scaring many a would-be user back to the safety of Zaxxon. Here's a look at a new service which makes sense of it all.

by Frank Lenk

It's pronounced *eye net*, and yes, it does watch over certain activities. However, its scrutiny is entirely benevolent... and the scrutinees are volunteers to a man.

It's mysterious, what...

At great personal peril, I've managed to discover some of the secrets of the enveloping presence of *iNet*. Coming up is the all encompassing guide to the ins and outs of a new type of computer communications service...

Aye In The Sky

The term *iNet* stands for "Intelligent Network". Knowing this key fact, you may nonetheless feel very little the wiser. I shall elucidate, starting with a bit of history...

In 1874, Alexander Graham Bell invented the telephone. Whether or not this was a smart move has yet to be seen. However, in 1876 the world's first long distance call was made between Brantford and Paris, Ontario and in no time Canada had been carved up by no less than seven major telephone companies, including Bell Canada, BC Tel, Alberta Government Telephones, and so on. In 1931 these regional operations connected themselves together to form the TransCanada Telephone System, or TCTS. In 1983 the TCTS modernized its image by assuming the imposing title of Telecom Canada.

Telecom Canada owns no property and employs no staff. A central staff in Ottawa is loaned from the member companies. This staff is responsible for some interesting stuff. In addition to

divvying up your long distance dollars among the member companies, Telecom is knee deep in computer communications.

In 1973 Telecom Canada introduced Dataroute, "the world's first nationwide digital data network", more recently enhanced by Dataroute International and Dataroute Multistream. In 1976 Telecom introduced Datapac, "the world's first commercially available packet switched network". This service too has been enlarged to include Datapac International.

Which brings us to the point of this story.

Datapac has become the common access doorway for virtually all of the commercial database services, such as Marketfax, InfoGlobe, BRS and many others. As the number of services and the number of users have both increased, this environment has become quite a hairy affair for the non-initiate.

There was an obvious need to make the whole ordeal much friendlier to the harassed working user.

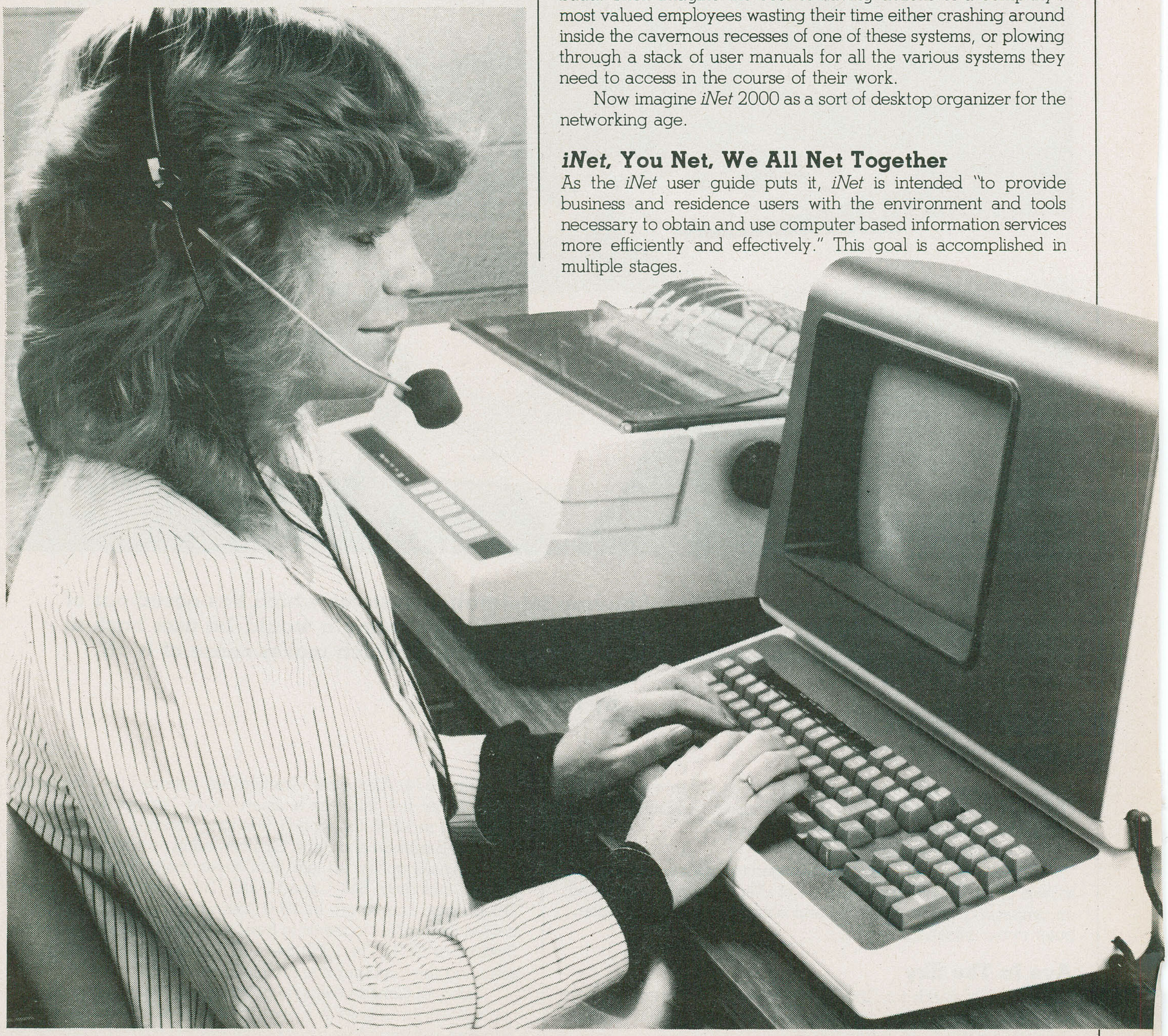
The *iNet* 2000 service was started up by Telecom Canada in answer to that specific need. We all know what inscrutable little beasts microcomputers can be. On line databases are a lot like that, only more expensive.

It's easy enough to spend a lot of three hundred baud time wandering around the many menus of the local BBS. Imagine the colossal confusion and cost of blundering through one of the real massive commercial data systems... even at twelve hundred baud. Then imagine the cost of having dozens of a company's most valued employees wasting their time either crashing around inside the cavernous recesses of one of these systems, or plowing through a stack of user manuals for all the various systems they need to access in the course of their work.

Now imagine *iNet* 2000 as a sort of desktop organizer for the networking age.

iNet, You Net, We All Net Together

As the *iNet* user guide puts it, *iNet* is intended "to provide business and residence users with the environment and tools necessary to obtain and use computer based information services more efficiently and effectively." This goal is accomplished in multiple stages.



As a bona fide *iNet* user you never have to log onto the individual data services. You dial up your local Datapac port number and enter any one of several valid *iNet* 2000 addresses followed by your *iNet* password. From then on you never leave the shelter of the cozy *iNet* environment.

You can select several types of interaction... essentially help levels. You can get one line prompts that indicate appropriate responses. You can have menus plus the prompt. Or you can opt out of these helpful modes, and have just a prompt character for entering direct commands.

One of the first things you'll do on *iNet* is to set up your user profile. This is managed through a menu that includes options such as your user interface display, your terminal and duplex mode, the language you wish to use... French or English... the character and line deletion symbols your system prefers to use, your screen width and so on. You can even set a command to be automatically executed whenever you log on, and an escape sequence that will get you out of any facility at any time and bring you back to the *iNet* prompt level.

The beauty of *iNet* is that all these features will stay constant, no matter how far you wander among the various data services.

The first menu is the origin. You can get back to it from any point in *iNet* by typing the command O.

ORIGIN MENU
1 Guide to using *iNet* 2000
2 *iNet* 2000 NEWs
3 List the National directory
4 List your Organization directory
5 List your Personal directory
6 Change/view your PROfile
7 Change your PAssword
8 *iNet* 2000 Messaging

Please enter a number to continue: 3

The national directory is a listing of information categories that can be searched through *iNet*. Rather than going through the tedious routine of poking around many individual databases, you can simply get *iNet* to tell you where to find the information you want.

National Directory
1 BUSINESS AND FINANCE
2 CONSUMER INFORMATION
3 EDUCATION, LAW AND HUMANITIES
4 ENTERTAINMENT AND TRAVEL
5 GEOGRAPHICAL DATA
6 GOVERNMENT AND POLITICS
7 INFORMATION SCIENCE/COMMUNICATIONS
8 SCIENCE AND TECHNOLOGY
9 SUBJECT CATEGORIES (ALPHABETICAL)
10 INET INFORMATION PROVIDERS
11 INET CLOSED USER GROUPS
12 INET SERVICES
No more

Please enter a number to see related categories: 1 100

All this gives one a lot of power and flexibility. You can search by a preset category... one through eight... or you can get a more detailed alphabetical category listing... option nine. Then again, you can go the more traditional route and simply list all the data services available... option ten. Option eleven has to do with those services which choose to restrict access to some degree. You can check those to see if you qualify.

Alternatively, you could have selected option four or five from the origin menu. These two selections allow you or your company to set up a custom directory of information categories, services or whatever. That way the stuff you use most often can be right at your fingertips. At present the custom menus have to be generated by the experts at Telecom. However, the next generation of system software... due on line in March... will let the user configure the custom menu unassisted.

Choosing any of the information categories drops you down to what is known as the category level, at which you get a more detailed breakdown. F'rinstance:

National Directory
1 BUSINESS AND FINANCE
Categories
1 ACCOUNTING
2 AGRIBUSINESS
3 BALANCE OF PAYMENTS
4 BANK RATE
5 BANKS
6 BONDS
7 BUSINESS
8 CASH FLOW
9 COMMERCE
10 COMMODITY MARKETS
11 COMPUTERS
12 CONFERENCES
13 CONSTRUCTION
14 CORPORATIONS
15 CURRENCY
16 DATA PROCESSING
17 ECONOMICS
Press ◄*CR*► to see more

Please enter a number to see related services: 10

Making a selection drops you to the service level.

National Directory
1 BUSINESS AND FINANCE
Category
10 COMMODITY MARKETS
1 FSIS BRS
2 HARF BRS
3 GRASSROOTS Grassroots
4* MARKETFAX STOCK SERVICE Marketfax Infoservices
5* MKTFX PROFESSIONAL STOCK SERVICE Marketfax Infoservices
6 PTSI, PTSL, PTSB, PTSW BRS
7 PTSP, PTSL, PTSB, PTSD BRS
8 PTSA BRS
9 PTSF BRS
10 PTSF BRS
11 STOCKPRICE Marketfax Infoservices
No more

Please enter a number to see the details pages: 11

We won't go through the next level... the detail level... in any depth. Suffice to say that selecting one of the above services will yield several screens full of information as to the full name of the service, the content, the cost, the proprietors and the type of search syntax one can expect from it. At the end of all this you are offered the chance to enter the command "access" if you wish to log on.

The experienced user is not forced to climb this tree to find a suitable branch of electronic wisdom. One could simply enter

list stockprice

at the command prompt and get exactly the same detailed information as above, bypassing all the menus. Or one could just as easily say

access stockprice

to immediately log on to the service.

And Furthermore

You may well ask what services are on *iNet*. Well, let's see here... There's *InfoGlobe,* the Globe and Mail's news retrieval service. There's *BRS*... the Bibliographic Retrieval Service... a compendium of dozens of individual databases. There are services run by Statistics Canada, by several Canadian universities and by the National Library of Canada. There are also a number of Videotex services, notably those operated by *Infomart.* Of course, you'll need a NAPLPS/Telidon graphics adapter to make any sense of these.

One of the major advantages of *iNet* is its consolidated billing. This means that all your access time for all of the member services is charged out on a single bill. You pay the usual rate for each, naturally.

The cost for *iNet* itself, by the way, is not particularly onerous. For the individual users there is a one time fee of twenty-five dollars. Corporate users pay fifty dollars. From then on the cost is fifty dollars per month for the organization, plus five dollars per individual user. Prime connect time goes for three dollars and sixty cents an hour, while non-prime time is two dollars and seventy cents. There is also a charge of thirty cents per thousand characters during prime time and twenty-two cents during non-prime time.

All these figures may seem a bit hefty until one compares them to the charges for any of the regular database services. Many of these run into upwards of fifty dollars per connect hour.

Another big advantage of *iNet* has to do with one of the other services offered by Telecom Canada, Envoy 100 electronic messaging.

This probably calls for a digression. Envoy 100 is a service accessible through Datapac, Dataphone, TWX and various foreign packet switched networks including the American Telenet and TYMNET systems. What it amounts to is electronic mail, connecting the sender to anybody hooked in to the growing computer web. EnvoyPost actually allows Envoy users to reach the few remaining non computerized neanderthals.

The relevance of all this is that Envoy messaging is accessible from within *iNet*. This gives the user electronic access to more than fourteen thousand individuals all over Canada.

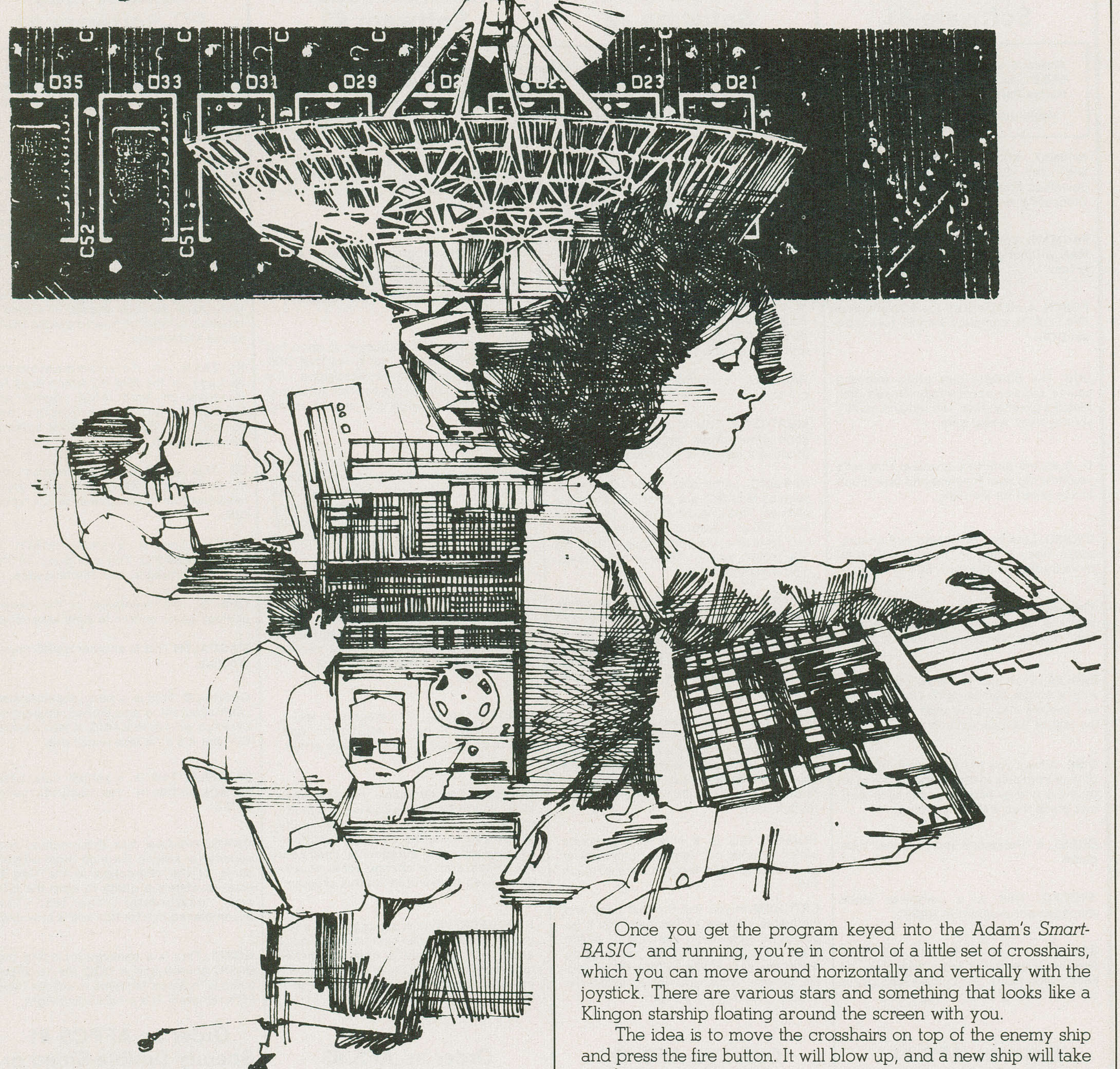
The Ottawa Connection

So far *iNet* is getting by on an interim approval from the CRTC. Its field trial started in July of 1982 and ran to February of 1984 without charging subscribers. The second phase, scheduled to run until August 1985, is the market trial. Assuming this goes well the CRTC should accredit *iNet* as an official service, as is the case with existing entities such as Datapac.

At present *iNet* is connecting about sixteen thousand users to between two and three hundred databases, operated by about thirty information providers. All these statistics should rise before the end of the trial period.

Tomorrow... the world! **CN!**

ADAM Star Patrol



Programming the Coleco ADAM is a lot like programming the Apple... with a number of very profound exceptions, as this game illustrates.

by Anthony DeBoer

Star Patrol is a fairly simple little game for the Coleco *Adam* that lets you fly around, zapping aliens. While it's nothing compared to a commercial cartridge game, like *Grog's Revenge,* it's not bad for eighty or so lines of BASIC.

Once you get the program keyed into the Adam's *SmartBASIC* and running, you're in control of a little set of crosshairs, which you can move around horizontally and vertically with the joystick. There are various stars and something that looks like a Klingon starship floating around the screen with you.

The idea is to move the crosshairs on top of the enemy ship and press the fire button. It will blow up, and a new ship will take its place.

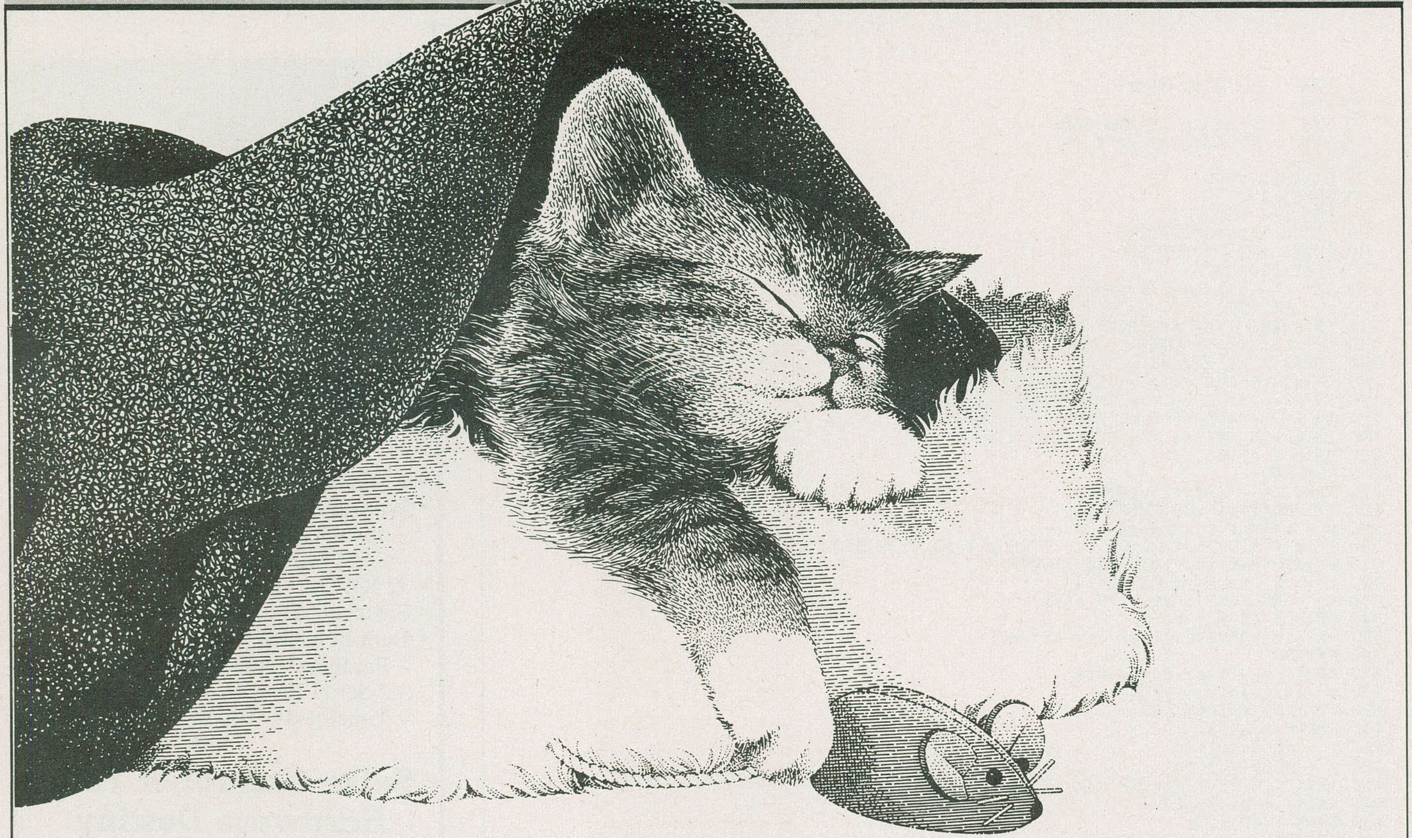
Displayed at the bottom of the screen are the number of enemy ships that have bitten the interstellar dust and the number of shots you've fired. Under that is your batting average. Keep it above about .4 and you're doing all right.

There is some fine print. Although Coleco SmartBASIC has some strong similarities to Applesoft BASIC, there are a few things Apple users should look out for if they're planning to adapt the program. Firstly, the Coleco handles its joysticks differently, so the section from lines 110 to 210 would need to be rewritten. Secondly, since the Coleco memory map is completely different, the section that sets up the shape table, in lines 730 to 780, would need changes.

```
  10 REM    --- Star Patrol
  20 REM
  30 REM     A Silly Little Game
  40 REM     by Anthony DeBoer
  50 REM
  55 HIMEM :50000
  60 GOSUB 470
  70 FOR i = 1 TO st
  80 GOSUB 110: GOSUB 270
  90 GOSUB 110: GOSUB 370
 100 NEXT: GOTO 70
 110 REM    --- player movement
 120 z = PDL(7)
 130 IF z AND  NOT tr THEN  GOSUB 620
 140 tr = z: z = PDL(5)
 150 XDRAW 2 AT xc, yc
 160 IF z = 1 AND yc > v THEN  yc = yc-v
 170 IF z = 2 AND xc < xl THEN  xc = xc+v
 180 IF z = 4 AND yc < yl THEN  yc = yc+v
 190 IF z = 8 AND xc > v THEN  xc = xc-v
 200 DRAW 2 AT xc, yc
 210 RETURN
 270 REM    --- enemy movement
 280 IF RND(1) < .05 THEN 350
 290 x = xe+ex: y = ye+ey
 300 IF x < v OR x > xl OR y < v OR y > yl THEN 350
 310 XDRAW 3 AT xe, ye: DRAW 3 AT x, y
 320 xe = x: ye = y: RETURN
 330 xe = FN r(xl): ye = FN r(yl)
 340 DRAW 3 AT xe, ye
 350 ex = v-v2*RND(1): ey = v-v2*RND(1)
 360 RETURN
 370 REM    --- star movements
 380 x = xx(i): y = yy(i)
 390 IF  NOT x THEN 430
 400 XDRAW 1 AT x, y
 410 x = q*(x-xc)+xc
 420 y = q*(y-yc)+yc
 430 IF x < v OR y < v OR x > xl OR y > yl THEN  x = FN r(xl): y = FN r(yl)
 440 xx(i) = x: yy(i) = y
 450 DRAW 1 AT x, y
 460 RETURN
 470 REM    --- initialization
 480 GOSUB 730
 490 st = 50
 500 DEF FN r(f) = v+f*RND(1)
 510 DIM xx(st), yy(st)
 520 TEXT: HGR: HOME: HCOLOR  = 3
 530 xc = 140: yc = 96
 540 q = 1.05: v = 5: v2 = v*2
 550 xl = 255-v2: yl = 159-v2
 570 FOR i = 1 TO st: xx(i) = FN r(xl): yy(i) = FN r(yl): DRAW 1 AT xx(i), yy(i
): NEXT
 580 DRAW 2 AT xc, yc
 590 GOSUB 330
 600 bd = 0: sf = 0: GOSUB 690
 610 RETURN
 620 REM    --- shot fired
 630 PRINT CHR$(7); : sf = sf+1
 640 IF ABS(xe-xc) > v OR ABS(ye-yc) > v THEN 690
 650 DRAW 4 AT xe, ye: PRINT CHR$(7); : DRAW 5
 660 XDRAW 3 AT xe, ye
 670 XDRAW 4 AT xe, ye: PRINT CHR$(7); : XDRAW 5
 680 bd = bd+1: GOSUB 330
 690 VTAB 20: PRINT
 700 PRINT "Hit: "; bd; " Shots Fired: "; sf
 710 IF sf THEN  PRINT "Batting Average: "; bd/sf; "          ";
 720 RETURN
 730 REM    --- set up shape table
 740 a = 50001
 750 READ d: IF d >= 0 THEN  POKE a, d: a = a+1: GOTO 750
 760 POKE 16766, 81: POKE 16767, 195
 770 SCALE  = 1: ROT  = 0
 780 RETURN
 790 DATA    5,0,12,0,14,0,24,0,50,0,109,0
 800 DATA    5,0
 810 DATA    36,252,147,42,109,45,222,51,46,0
 820 DATA    56,63,63,55,55,55,46,46,110,73,73,73,44,44,36,39,39,63,63,4,44,223
,51,28,7,0
 830 DATA    219,35,39,39,39,44,37,39,36,37,45,53,53,53,45,37,44,44,44,36,45,46
,54,54,45,46,53,55,62,54,53,53,53,53,53,55,55,39,60
 831 DATA    63,62,62,62,38,39,39,55,55,55,63,60,60,36,39,63,39,44,5,0
 840 DATA    60,60,60,60,44,45,45,37,39,39,39,45,45,45,53,54,54,46,45,45,55,55,
55,55,55,62,55,55,62,36,60,63,39,37,37,0
 850 DATA    -1
```

CN!

A Small Cat for CP/M



Keeping a handle on a growing collection of disks can be a bit of a task. If you've already worn out the control and C keys on your system searching through countless directories, consider implementing this simple program.

by Steve Rimmer

Although I haven't enumerated them recently, I think there must be almost a thousand disks in this place. Now, that isn't as awesome as it seems... some are for the typesetter, some are five inch ones, quite a few are cropping up for the three inch systems, but that still leaves me with several hundred to plough through if I want a file I haven't touched in a couple of months.

You probably don't label your disks any better than I do. They all have hieroglyphics on them. I have the WS series... I seem to recall they were mostly text files. The ones labelled MAC are likely assembler files. Four or five marked MBC turned out to be Microsoft BASIC source disks... and so on.

One can easily spend hours popping disks in and out of one's drives looking for a particular file. If you've installed ZCPR on your computer... see the article on the hacking thereof in the last edition of Computing Now!... and have thereupon taken to making use of the higher user areas your task may be even wilder.

A cataloging program can make all this a bit simpler. It takes a while to set up a catalog but, once you've done it you can search through a catalog file or a few pages of paper rather than several boxes of disks.

Cat o' Nine Tails

The CAT program is pretty straight up to use. You put it on a disk... with a fair bit of room... in drive A: and invoke the program. You can give it a file parameter, such as

```
ACAT *.ASM
```

and create a catalog of a specific sort of file.

Heave a disk in drive B: and give CAT a name for it. This should consist of the date and whatever inscriptions you have etched on the outside of the disk. Hit return and CAT will create a file called CATALOG.CAT on drive A: containing your header and the catalog of the disk.

Yes, this is incredibly real, but, wait, there's more. If you run CAT again with a different disk in drive B: and give it a new header, the CATALOG.CAT file will contain the directory of both disks. In fact, every time you run CAT the file will be updated with a new directory.

When you're done you should have a long file with directories of all your disks and suitable means for identifying them.

If you wanted to find the file FROGNOSE.ASM you could

```
;       CATALOGCATALOGCATALOGCATALOGCATALOGCATALOG
;
;       Disk cataloging program for
;       CP/M Copyright (c) 1985 Steve Rimmer
;
;       Can also be used for enumerating unicorns
;       if you change all the code
;
;       CATALOGCATALOGCATALOGCATALOGCATALOGCATALOG

;----   ASSORTED DEFINES

VERS    EQU     6       ;VERSION
SUBVERS EQU     9       ;SUBVERSION... CALL THE RCMP

CR      EQU     13      ;CARRIAGE RETURN
LF      EQU     10      ;LINE FEED
TAB     EQU     9       ;TAB
CLS     EQU     26      ;CLEAR SCREEN
EOF     EQU     0       ;FILE PADDING CHARACTER
BDOS    EQU     0005H   ;CALL HERE FOR PIZZA
DMA     EQU     0080H   ;THE DMA BUFFER
COMTL   EQU     005CH   ;THE COMMAND TAIL
FL$BLK  EQU     005CH   ;THE FILE CONTROL BLOCK
DRIVE   EQU     3       ;1 FOR A, 2 FOR B

        ORG     0100H

;----   SET UP A LOCAL STACK
        LXI H,0 ! DAD SP ! SHLD STACK ! LXI SP,STACK

;---    AND HOP OVER THE FIXED STUFF
        JMP     START

;----   BUFFERS AND SUCHLIKE
FCB     DB      DRIVE,'???????????'
        DB      0,0,0,0,0,0,0,0             ;DISK DIRECTORY FCB
FCB$CAT DB      0,'CATALOG CAT'             ;FCB FOR CATALOG.CAT
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
FCB$BAK DB      0,'CATALOG BAK'             ;FCB FOR CATALOG.BAK
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
        DB      0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
FCB$REN DB      0,'CATALOG CAT',0,0,0,0 ;FCB FOR RENAME
        DB      0,'CATALOG BAK',0,0,0,0
N$DIR   DS      2                           ;BUFFER FOR POINTER
CON$BUF DS      60                          ;CONSOLE BUFFER

;----   WRITE THE BUFFER AND COPY THE .BAK FILE TO .CAT
WR$BUF:
        LXI H,FCB$CAT ! LXI D,FL$BLK ! MVI B,12
        CALL MOV$STR ! LXI H,FL$BLK ! CALL FL$OP
        CALL    DELETE          ;DELETE OLD BAK FILE
        CALL    SEARCH          ;SEE IF CAT FILE EXISTS
        CPI OFFH ! JZ NO$BAK    ;SKIP RENAME IF NO CAT FILE
        CALL    RENAME          ;RENAME EXISTING FILE TO BAK
NO$BAK  CALL    CREATE          ;CREATE NEW CAT FILE
        CALL    COPY            ;COPY BAK TO CAT

        LXI D,DIR$BUF           ;COPY BUFFER TO CAT
WR$LP   PUSH D ! CALL SETDMA ! CALL WRITE
        POP D ! LXI H,128 ! DAD D ! XCHG
        MVI A,EOF ! MVI B,128 ! CALL SCAN
        JNC WR$LP ! CALL SETDMA
        CALL WRITE ! CALL CLOSE

        RET                     ;BACK TO CALLER

;----   COPY BAK TO CAT ONE SECTOR AT A TIME
COPY:
        CALL OPEN ! CPI OFFH ! RZ       ;OPEN FILE
        LXI D,DMA ! CALL SETDMA         ;SET DMA
COP$LP  CALL READ ! CPI 0 ! RNZ         ;READ A SECTOR
        CALL WRITE ! CPI 0 ! JNZ WRT$ERR        ;WRITE IT
        JMP COP$LP                      ;LOOP TIL DONE

;----   ZERO THE APPROPRIATE FIELDS IN FCB IN HL
FL$OP:
```

```
        MVI A,0 ! LXI D,12 ! DAD D ! MOV M,A
        INX H ! INX H ! MOV M,A ! LXI D,18
        DAD D ! MOV M,A ! RET

;----   SEARCH A SECTOR FOR EOF, C=1 IF ONE IS FOUND
SCAN:
        PUSH D ! XCHG
SCN$CN  CMP M ! JZ SCN$F ! INX H ! DCR B ! JNZ SCN$CN
        POP D ! STC ! CMC ! RET
SCN$F   POP D ! STC ! RET

;----   SEE IF CAT FILE EXISTS ON DISK
SEARCH:
        LXI H,FCB$CAT ! CALL FL$OP ! MVI C,17
        LXI D,FCB$CAT ! CALL BDOS ! RET

;----   READ A SECTOR FROM THE BAK FILE
READ:
        MVI C,20 ! LXI D,FCB$BAK ! CALL BDOS ! RET

;----   OPEN THE BAK FILE TO READ FROM IT
OPEN:
        LXI H,FCB$BAK ! CALL FL$OP ! MVI C,15
        LXI D,FCB$BAK ! CALL BDOS ! RET

;----   RENAME EXISTING CAT TO BAK
RENAME:
        MVI C,23 ! LXI D,FCB$REN ! CALL BDOS ! RET

;----   DELETE OLD BAK FILE
DELETE:
        MVI C,19 ! LXI D,FCB$BAK ! CALL BDOS ! RET
;----   CREATE NEW FILE
CREATE:
        LXI H,FL$BLK ! CALL FL$OP ! MVI C,22
        LXI D,FL$BLK ! CALL BDOS ! RET

;----   WRITE TO NEW FILE
WRITE:
        MVI C,21 ! LXI D,FL$BLK ! CALL BDOS ! RET

;----   CLOSE NEW FILE
CLOSE:
        MVI C,16 ! LXI D,FL$BLK ! CALL BDOS ! RET

;----   MOVE STRING IN H TO D, LENTH IN B
MOV$STR:
        MOV A,M ! STAX D ! INX D ! INX H
        DCR B ! JNZ MOV$STR ! RET

;----   FILL STRING IN H WITH A, LENTH B
FILL:
        MOV M,A ! INX H ! DCR B ! JNZ FILL ! RET

;----   PRINT STRING IN H 'TIL EOF
PR$BUF:
        MOV A,M ! CPI EOF ! RZ ! MOV E,A
        MVI C,2 ! PUSH H ! CALL BDOS
        POP H ! INX H ! JMP PR$BUF

;----   DO 16 BIT COMPARE
COMP16:
        MOV A,D ! CMP H ! JNZ COMP1
        MOV A,E ! CMP L ! JNZ COMP1
        STC ! RET
COMP1   STC ! CMC ! RET

;----   IN LINE PRINT UTILITY. SEE "UTILITY BLUES"
ILPRT:
        XTHL
ILPLP   MOV A,M ! ORA A ! JZ ILPRET
        PUSH H ! MVI C,2 ! MOV E,A ! CALL BDOS
        POP H ! INX H ! JMP ILPLP
ILPRET  XTHL ! RET

;----   SET DMA
SET$DMA:
        MVI C,26 ! CALL BDOS ! RET

;----   SEARCH DIRECTORY FOR FIRST
```

A Small Cat for CP/M

```
S$FIRST:
        MVI C,17 ! LXI D,FCB ! CALL BDOS ! RET

;----   SEARCH DIRECTORY FOR NEXT
S$NEXT:
        MVI C,18 ! LXI D,FCB ! CALL BDOS ! RET
;----   FILE DIRECTORY BUFFER WITH EOF
INT$BUF:
        MVI B,16 ! LXI H,DIR$BUF
INT$LP  PUSH B ! LXI B,-1 ! MVI A,EOF ! CALL FILL
        POP B ! DCR B ! JNZ INT$LP ! RET

;----   GET A STRING FROM THE CONSOLE
GET$TXT:
        MVI C,10 ! LXI D,CON$BUF ! MVI A,40
        STAX D ! CALL BDOS ! LXI H,CON$BUF+2
        LDA CON$BUF+1 ! LXI D,0 ! MOV E,A
        DAD D ! MVI A,CR ! MOV M,A
        MVI A,LF ! INX H ! MOV M,A
        LXI H,CON$BUF+1 ! INR M ! INR M ! RET

;----
;----   THIS IS WHERE THE MAIN PART OF THE PROGRAM STARTS
START:
        CALL    INT$BUF ;CLEAN OUT THE BUFFER
        LDA     COMTL+1 ;IS THERE
        CPI     ' '     ;...AN ARGUMENT?
        JZ      NO$ARG  ;IF NO, PASS
        LXI     H,COMTL+1
        LXI     D,FCB+1
        MVI     B,11    ;IF THERE IS...
        CALL    MOV$STR ; ...MOVE THE STRING

NO$ARG: CALL    ILPRT
        DB      CLS,TAB,TAB
        DB      'Wombat Brothers Disk Catalog Program'
        DB      CR,LF,TAB,TAB
        DB      'Galactic revision ',VERS+'0','.',SUBVERS+'0','.'
        DB      CR,LF,TAB,TAB
        DB      '  Copyright 1984 (c) Steve Rimmer   '
        DB      CR,LF,LF
        DB      'Lay a name for this disk on me: ',0

        LXI D,DMA ! CALL SET$DMA        ;SET UP DMA BUFFER
        LXI H,ODOAH ! SHLD DIR$BUF      ;INSTALL LEADING CRLF

        CALL    GET$TXT                 ;GET NAME FOR DISK
        LXI H,CON$BUF+2 ! LDA CONBUF+1
        MOV B,A ! LXI D,DIR$BUF+2
        CALL MOV$STR ! XCHG ! SHLD N$DIR

        CALL S$FIRST ! CPI OFFH ! JZ NO$FRST

DR$LP   RLC ! RLC ! RLC ! RLC ! RLC
        LXI D,0 ! MOV E,A ! LXI H,DMA
        DAD D ! INX H ! XCHG            ;GET POINTER IN D
        LHLD N$DIR ! XCHG ! MVI B,11

        CALL    MOV$STR         ;MOVE ENTRY INTO TABLE
        LHLD N$DIR ! LXI D,11 ! DAD D
        MVI B,5 ! MVI A,' ' ! CALL FILL

        LHLD N$DIR ! LXI D,16 ! DAD D
        SHLD N$DIR              ;BUMP POINTER

        CALL S$NEXT ! CPI OFFH ! JNZ DR$LP

        CALL    ILPRT
        DB      CLS,CR,LF,LF,TAB
        DB      'Directory for the disk ',0

        LXI     H,DIR$BUF+2     ;POINT INTO BUFFER
        CALL    PR$BUF          ;SHOW BUFFER
        CALL    WR$BUF          ;WRITE BUFFER TO DISK

QUIT:
        LHLD STACK ! SPHL ! RET

;----   ERROR FOR NO MATCHING FILES ON THE DISK
NO$FRST:
        CALL    ILPRT
        DB      CR,LF,LF,TAB,TAB
        DB      'Ack... it''s the cosmic vacuum',0
        JMP     QUIT

;----   DISK WRITE ERROR ON CATALOG.CAT
WRT$ERR:
        CALL    ILPRT
        DB      CR,LF,LF,TAB,TAB
        DB      'Ack... it''s a nasty write error',0
        JMP     QUIT

        DS      60      ;LOCAL STACK
STACK   DS      2       ;STASH FOR OLD STACK POINTER

DIR$BUF EQU     $       ;WHERE THE BUFFER BUILDS ITSELF

        END
```

simply inhale the CATALOG.CAT file into WordStar and search for that name. Alternately, if you're into primitive technology, you could print the file out and search through it by hand... or, rather, by eye.

The CAT program included here can handle a catalog file of any size, limited only by the amount of disk space you have available to manipulate it with. Bear in mind that it creates a BAK file. It will file any sort of CP/M disk your drives can read.

Here, Pussy

There's nothing particularly weird about the source code for CAT. It uses no macros and, as such, can be assembled with either MAC or ASM. You'll notice a fair number of dollar signs in the labels... these are an attempt to make them more readable, and don't actually mean a great deal.

The important bits about CAT are in its disk file handling... it does quite a lot of this, for obvious reasons. Much of the code here can be lifted for other applications once you get into how it does its stuff.

There are, in fact, two things happening in CAT. The first bit gets a directory listing of the current disk in drive B: and stores it in a buffer. The latter half adds this listing to whatever is already in CATALOG.CAT.

Getting a directory of a disk is actually surprisingly simple, especially if you don't really care whether it's sorted or tagged with file sizes. I've omitted these details here to keep the code down to a manageable hugeness. The October 1984 edition of Computing Now! has a sorted directory listing you can integrate into this troll if you feel up for it.

The CP/M BDOS embodies two calls for getting a directory listing happening... they also serve to inquire as to the existence of files, as we'll see. Call seventeen looks for the first instance of a file and call eighteen the next. You can keep calling the latter until you run out of files that match what you've asked for.

Obviously, if you specify a real file name there will only be one instance of it in a directory. The search functions are usually used with *ambiguous* file names, or, in humanspeak, names with wild cards in them.

If you ask for something like *.ASM CP/M will create a file control block with the string ????????ASM in it. The question marks are ambiguities... any character can match with one. If you issue the command DIR you are actually saying

DIR ???????????

To make the search commands work you must set up an FCB with the name field filled in with your file name and whatever ambiguities you have in mind. Since specifying a file name when you

call CAT will cause CP/M to create a padded file name string for you... question marks and all... all you really have to do to get this together is copy the string from where CP/M puts it at 005CH into your own FCB.

The FCB in question is up at the top of the file, labelled... yes... FCB:. The first byte, DRIVE, will hold the drive code for the disks you want to catalog. We'll eventually move this string back down to 005CH, which is the default FCB for most CP/M functions.

The CAT program starts out by clearing out a large chunkof RAM immediately above itself, starting at the label DIR$BUF. It fills in this buffer with the EOF character... I've used a null here, but anything unprintable will do. A lot of these EOF's will turn up in the CATALOG.CAT file, so be sure to choose something your printer will ignore.

The first two bytes of this buffer are filled in with a carriage return and a line feed to make the listing look pretty. A pointer into the buffer is maintained in N$DIR, up at the top of the file.

Having filled in the buffer with a directory listing the program will print it to the screen and then call WR$BUF, which handled all the subsequent disk file operations.

Diskonnected

The WR$BUF routine does two things. To begin with, it deletes the existing BAK file if one exists. The BDOS delete function is such that we can delete the thing whether it exists or not. Next it will use the search for first function to see if CATALOG.CAT exists. If it does, it will rename it to CATALOG.BAK. Notice the buffer called FCB$REN. It's a funny sort of FCB, with two names in it. The first sixteen bytes contain the file that exists and the second the file it will become after the call.

The idea here is to append data to the file CATALOG.CAT. Sadly, CP/M doesn't provide a really reliable way of doing this. Reading through the file and then writing to the end of it is liberally frought with nasties. What we're actually going to do is to copy the old file into a new file and then copy the new data onto that. It'll look to CP/M like one long continuous write operation.

Having copied the data from the old file into the new one... the file's still open, so the record count is now pointing to end of the new file... we can send the data in the buffer into the new file as well.

Up until now we've been copying the data from the old file into the new file one sector at a time. This is not a particularly slick way of doing it... it should be buffered... but it's fairly easy to code. Now, however, we have a whole buffer full of data to write, which changes the nature of the problem.

Data flows on and off the disk through the DMA buffer. The DMA buffer is a block of a hundred and twenty eight characters which is anywhere we have previously said it is through means of the DMA setup BDOS call, function twenty six. In the case of the file copying routine the DMA was set to point to 0080H... CP/M maintains an otherwise unused buffer there for disk fudging... and the program alternately wrote and read using this space.

Moving the buffer onto the disk is a bit of a different trip. We could copy it into the DMA space in blocks of a hundred and twenty eight characters and write it as we have been doing, but it's a lot easier to simply move the DMA pointer through the disk directory buffer... ol' DIR$BUF, as you'll recall... bumping it up by a hundred and twenty eight bytes after each write operation.

Having filled the buffer with EOF's a while ago, it's fairly easy to tell when we've written the whole thing. The first block which contains an EOF is the last one to be written.

By the way, the mnemonic EOF usually represents control Z, the CP/M end of file character. In the case of this program I haven't used a control Z because quite a number of these things will usually find themselves trapped in the middle of CATALOG.CAT as it builds up. Among other things, they clear the screens of most computers if they're typed.

End Of File

Having gotten CAT working you'll probably find no end of possibilities in modifying it. There is quite a lot of stuff that can be added to this code. If your system has a clock you should include that in the catalog file. You might want to sort and add size tags to the directory listings.

Assembler programs are made for hacking with.

I think that what's really called for here is one unified ultimate utility. I mean, you could put it all in there... give it the right command prompts and it will be a disk directory listing, a catalog, an assembler, a telecommunications terminal, a spread sheet, a list sorter and with a couple of tags and options will end disease, hunger, pestilence, oppression and day time television.

Actually, we were going to do such a program but the source code was a little long so we shelved it. No one really wants to type in a huge listing. **CN!**

Searchmart

On line data bases are everywhere. Here's a look at one of the companies providing information over the wire.

by Frank Lenk

There's a new marketing venture born every minute, or so it seems. Still, the approach being promoted by Searchmart Corporation is kind of unique. They're marketing computer products by computer... hmm. Yes, it sounds almost too obvious.

Of course, that's not the whole story behind Searchmart, a company that is attacking the market on a fairly broad front. The general idea involves running commercial databases that offer free access to the public. I paid the Searchmart folks a visit at their local office, and had a tour of their two online services.

Software Service

Searchmart is a sort of hybrid American and Canadian effort, the founding genius and president being one Victor Gruneau, who is based in North Palm Beach, Florida. Gruneau hit upon the idea of using on line computer services to aid advertisers. The first of his

SMITH-CORONA JUST NARROWED YOUR CHOICE OF COMPUTER PRINTERS DOWN TO FIVE.

Introducing the personal computer printers you've been waiting for. From Smith-Corona.

Each one is remarkable. Each one affordable. Each one designed to meet very particular needs. Starting with the need for speed.

They're fast workers. Put words and numbers on paper at the crisp rate of 140 characters per second with the new Smith-Corona D-300 Dot Matrix Printer. Or as fast as 120 characters per second with the D-200 and D-100.

Compatible partners. The new Smith-Corona printers work perfectly with most major computers. Including IBM, Apple, Commodore and more. Four of these printers feature a Dual Interface to ensure wide compatibility. The fifth, the Smith-Corona Fastex 80 Dot Matrix Printer, offers Parallel Interface.

High quality printers. As you can see, the new Smith-Coronas deliver outstanding results. Even at high speeds. **This Dot Matrix print out is as close to letter quality as you can get.** While the L-1000 Daisy Wheel model produces documents that satisfy your most critical needs. Naturally, since document sizes vary, our paper capacity varies too. From standard 8½" computer paper to 15" widths. (And even smaller sizes when sheet-fed.)

And a reliable investment, all down the line. Because we're Smith-Corona, you get reliability and workmanship second to none. Our national service network is as near as your phone.

So why waste time? To get down to serious computing around your home or office, look into the new line of personal printers from Smith-Corona. Your choice will be easier than you think.

For more information about the full line of new Smith-Corona printers, call your nearest authorized Smith-Corona dealer. Or fill in and mail the coupon below.

CN 02/85

☐ Please send me more information about Smith-Corona printers for in-home use.

☐ Please send me more information about Smith-Corona printers for office use.

NAME: ________

COMPANY NAME: ________

BUSINESS ADDRESS: ________

CITY ____ PROV. ____ POSTAL CODE ____

TYPE OF BUSINESS ________

Send to: Bryce Buskard

SMITH CORONA

DIVISION OF SCM (CANADA) LIMITED
440 TAPSCOTT ROAD, SCARBOROUGH, ONT.
M1B 1Y4 (416) 292-3836

Circle No. 6 on Reader Service Card

brainchildren now operates from his Florida location. However, Searchmart's newer Rexdale office is taking over much of the company's managerial burden, including the development of the required software.

Of Searchmart's two existing services, the one with the most general appeal is the on line software library. Their brochure calls it "the first electronic software locator service that offers software buyers free access to detailed information about the tens of thousands of software products that are currently on the market." That basically sums it up, although I should add that since the service is just getting underway the part about "tens of thousands" must be taken somewhat more in the poetic than in the literal sense.

The way it works is this. Searchmart has set up this database. Essentially the whole thing looks just like a massive bulletin board service, though it does run on a multi-processor GEAC 6000. Searchmart approaches the software publishing companies all over the States and Canada, and gets them to pay for advertising space on the system. Users phone the system... at 1 305 845 6466... and read all the information, thereby solving all their software queries and moving all the advertisers' wares.

And we all live happily ever after...

Actually, there's a sort of bootstrapping process involved here. Nobody is going to advertise on this newfangled contraption until it has been thoroughly demonstrated that people are actually going to access it. Nobody is going to access this Florida phone number until they have reason to believe that something moderately interesting is to be found on the other end.

Searchmart has several solutions to this seemingly insoluble dilemma. First off, the long distance problem is easily solved by setting up some branch operations. A Toronto line should be active early in the new year.

As to the chicken and the egg thing... what Searchmart is doing is a lot like "priming the pump". The company chooses interesting software prospects and sets up its own dummy listings for them. This fills up a lot of the vacant space on the system, and gives callers something to read. While callers read these preliminary texts, the system collects statistics as to who read what and how often.

By the time I spoke to Peter Bowers, President of the Canadian operation, he was able to hand me about eleven pages of finely printed names... "some of the many thousands of software buyers who accessed the online software library during the first weeks after going online", it said. The list reads like a who's who of American Commerce... everybody from AT&T, in New Jersey, to Yavapai Community Hospital in Prescott, Arizona.

Searchmart shows this hefty prospectus to the software companies, and... in principle, at least... goes on to achieve greatness.

Searching

The library database is set up to make things easy on the caller. It queries one for background on the type of computer being used, the operating system and the type of software the caller is after. This should let even relatively confused browsers find just the product they need.

Let's say one wanted to get a word processor for an IBM PC... God knows why. Having logged on to the on line software library one would get some introductory verbiage and then the following menu.

Criteria Menu Computer Types

1 Micro Computers	**2 Mini Computers**
3 Mainframe Computers	**4 Not Relevant to Search**

Enter the type of computer for which you desire software ► 1

Which would be followed by

Micro Hardware

1 Altos Computers	**2 Apple Computers**
3 Commodore	**4 Cromemco**
5 Digital	**6 Heath/Zenith**
7 Hewlett Packard	**8 IBM Micro Computers**
9 NEC Computers	**10 Northstar**
11 Onyx Computers	**12 Osborne**
13 Tandy/Radio Shack	**14 Televideo**
15 Texas Instruments	**16 Not Relevant to Search**

Enter the hardware or compatible computer model ► 8

This in turn would lead to

IBM Operating Systems

1 Basic 2.0	**2 BMAS**
3 BRADS	**4 CP/M 86**
5 M DOS	**6 MP/M 86**
7 MS DOS/PC DOS	**8 ROS**
9 UCSD P	**10 VM/CMS**
11 Not Relevant to Search	

Enter the operating system ► 7

Next, one would get a category listing. Rather than bore you with the whole thing, I'll merely say that it includes all the traditional fields of endeavor, amusements and games, data communications, graphics, general business... Ah hah! General business it is.

Sure enough, category four... general business... steered me to yet another menu.

General Business Sub Category

1 Accounting	**2 Administrative/Executive**
3 Data Bases	**4 Economic/Econometrics**
5 Financial Management	**6 Graphics**
7 Laboratory Systems	**8 Mailing Lists**
9 Office/Plant Management	**10 Personnel Management**
11 Sales Analysis	**12 Secretarial**
13 Standard Spread Sheets	**14 Word Processing**
15 Others	**16 New Products**

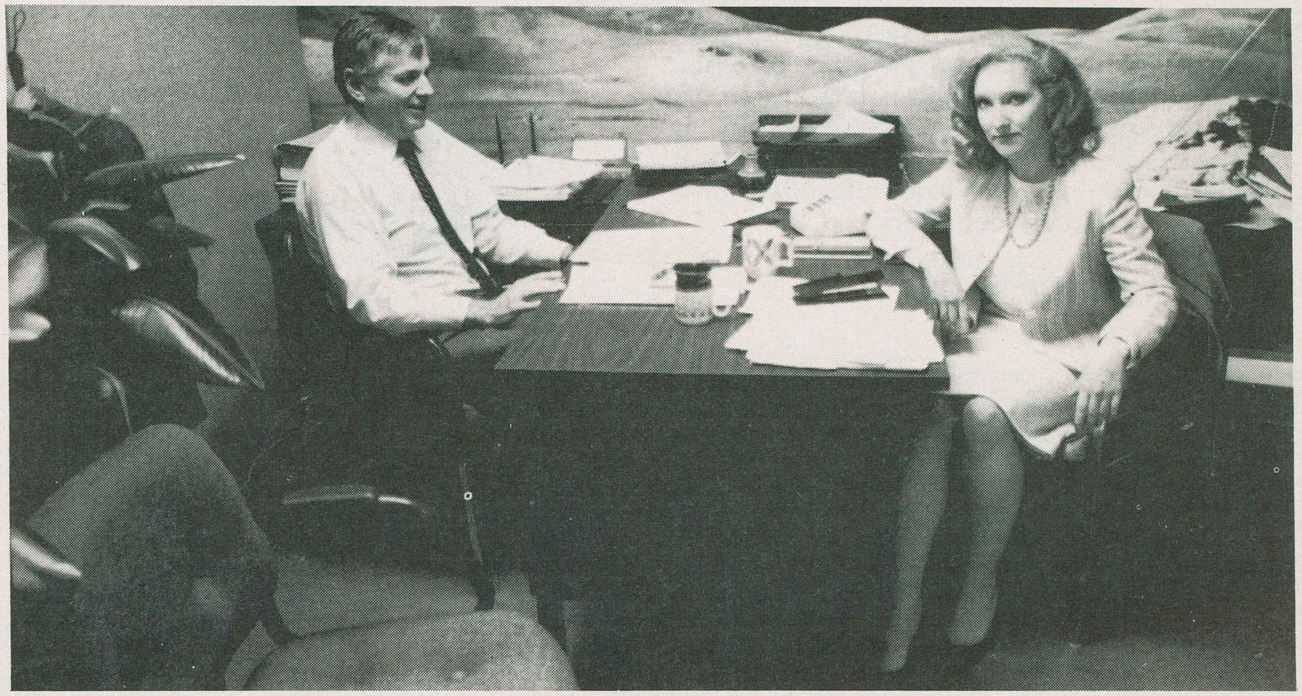
Enter category number or "H" for help.
► 14

And sonofagun, there it was.

Picking category fourteen caused the system to disgorge a listing of thirty-six pieces of software, in no apparent order... everything from *Volkswriter Deluxe* to *Proscript Scientific,* in fact. I had no way of knowing which were genuine listings and which merely place holders, but the two I picked at random... *Samna Word* and *The Word Plus*... were both well documented.

The detailed listings included two standardized pages, listing hardware requirements, operating system, language, the developer's address, prices, and a contact for purchasing enquiries. Following this came two pages of advertising, inserted by the advertiser or the publisher.

Thus, for *Word Plus*, from Oasis Systems, I was informed that:



This package is designed to check 10,000 words of text in less than minutes. Its vocabulary contains over 45,000 words. The program will locate and correct spelling errors while it automatically changes corrections to match capitalization and possible endings. The word plus includes a feature to automatically hyphenate words within text files. word plus analyzes documents by compiling a list of words and showing how often each one is used. this feature helps reduce redundancy. the word plus program allows the user to expand its vocabulary. the dictionary feature is indexed for rapid access. this feature, combined with other programs included with the word plus, locates rhyming words,

Enter "F" to continue.
► F

and solves crossword puzzles, word jumbles and anagrams.

As you can see, it turns out that this is not exactly the beast I was after, being a mere spelling checker and not a real word masher. Ah well, back to the old menu screen.

The library offers some commands to make the searching a bit faster. At any command prompt you can move backward to the previous screen, forward to the next, right back to the hardware critical menu, or just back to the previous menu. You can also inform a software vendor of your interest.

The Media are the Messages

The other service run by Searchmart is as esoteric as they come, but nevertheless well worth a quick browse. The trick is to get on it at all, since this one is not open to the general public. I was especially nice to the people in the Rexdale office, and they consented to give me a peek.

Known as *specialized media on line*, this database is actually a service especially designed for large advertisers. The idea is that the major ad agencies need up to date and comprehensive stats on the many specialty publications littering the newsstands. Only with this information can they make intelligent choices as to what should be advertised where, and how much said ads are really worth.

Although access to this service is restricted, it is... once again... free to the users. Its operation is funded by charging magazine publishers to place their listings. This lets the publishers set up a planned presentation that can efficiently pull in the big advertisers.

Of course, it's not all hype. Most of the listing format is based on hard statistical data. Thus the advertisers get the facts they need, while the publisher gets a chance to throw in a bit of a sales pitch. Furthermore, everyone saves time and effort since advertisers can easily discard publications that are obviously not suited for plugging their product.

The media database is much more sophisticated than the BBS like software library. There are three modes of operation, the display mode, the calculation mode and the presentation mode. These functions give the same sort of data manipulating ability as you'd get in any good data management package.

The display mode lets the advertiser choose a magazine title and the type of profile to be generated. Profiles may be generated on the basis of various parameters... geographically, by advertising rates, by circulation data, by editorial profile, by market served, and so forth.

The calculation mode lets the advertiser specify a magazine or magazines, a target audience, the frequency of the planned ads, and then calculate the proportion of each magazine's audience that the ad would actually be aimed at, the effective cost per thousand target readers and total campaign cost, or the sources and dates of the statistics upon which the calculations are based. Furthermore, the system makes it easy for the advertiser to play around with the base parameters and generate what if profiles for various magazines, ad sizes, and so forth.

Finally, the presentation mode is where the publishers get to throw their real heavy duty sales pitches. In this mode, the system displays blocks of text set up by the publishers themselves. Often this will include extra data, such as page usage breakdowns. At least one publisher to our knowledge had indulged in a bit of online character graphics.

Access to the media service is by invitation only, so unless you happen to work for a major advertising agency, you'll probably never get a chance to fool around with any of this stuff. **CN!**

WordStar Backrub

Many computers are actually designed with WordStar in mind. Quite a few others are not, and a few seem to have been created by someone who really hated this faithful servant of mankind. Here's a quick fix for one of its most persistent problems, the unavailablility of a delete key on many computers.

by Steve Rimmer

When the cosmic centurions handed down WordStar from nirvana... I know, that theory is contested by some authorities... they laid quite a number of hidden teeth in its soft furry little face. Some of these were not to make themselves known for years.

The most troublesome aspect of applying WordStar to many CP/M based systems is in getting it to deal with the backspace key properly. The CP/M backspace character is control H. Under CP/M this little gaffer backspaces and erase the character it has backspaced over. WordStar interprets it differently, making it simply a command to drag the cursor one column to the left.

This is a downer for a number of reasons, not the least of which is that control H isn't even the right command for moving the cursor... most normal humans with the allotted number of thumbs and fingers use control S. The second serious hassle is that many keyboards aren't equipped with a convenient way to generate the character which WordStar wants to use as a proper... destructive... backspace.

The mystic character is 7FH, the RUB code.

We've looked at a number of solutions to this problem for specialized situations, most of them having to do with running WordStar on an Apple. However, as quite a number of other computers are no better at doing rubs than fruits are, we are now going to check out an elegantly simple solution to this quandry which has been vexing mankind for centuries.

Make that milli-centuries.

New Patches

Because it was designed to be adapted to a variety of computers and terminals with widely varying characteristics, WordStar maintains a bit of itself called the *terminal patch area*. This thing allows for most of the I/O functions which WordStar performs with internal routines or through CP/M BIOS calls to be replaced with code written by actual humans such as you or I or Orfid the white dwarf down the block.

In the case of this little fix, the code is actually pretty trivial. In fact, it has to do two things. First off, we must initialize the patch when WordStar first boots. Thereafter, WordStar's input calls must be confused so as to be spewed through the patch rather than directly to where it thinks it's supposed to be going.

Initializing all this is fairly straight up. WordStar provides a three byte space called INISUB which normally holds the code

INISUB: NOP ! NOP ! RET

It will run this clever little routine every time it's booted, just prior to turning into a word processor. This bit of code presently does nothing, and does it very well.

We're going to replace this thing with a jump instruction which points to the patch initialization code. Thus, every time WordStar boots it'll set up the patch.

There's another one of these things which is called when WordStar wants to suck a character in from the keyboard. It goes

UCONI: NOP ! NOP ! RET

We'll replace it with a jump to the actual code which is going to form the patch.

Finally, there is an area in WordStar which is specifically designed to hold patches. There's nothing in it normally. It's cool with WordStar if we stick some code there. The patch area is called MORPAT.

Orfid The Dwarf Strikes Back

The program in listing one is an assembly language routine that makes all this happen. If you've done any assembly language programming before this one will look a bit strange. For one thing, it ORG's in three places.

When you set up an ORG you are telling the assembler to create a hex file which specifies that the stuff after the ORG should start at the specified address. By ORGing the first bit of the code at MORPAT, we are telling the assembler to assemble code to go there.

Turning the resultant hex file into a COM file would be disasterous. COM files have to start with an ORG of 0100H. However, as you'll see we won't be doing it that way.

There are two bits to the code. The first one, SETPAT, initializes the patch. It will be called once when WordStar boots through the jump we'll put at INISUB. When WordStar wants to get a character in it calls the CP/M BIOS directly, which it can locate programmatically.

SETPAT does about the same thing. It locates the BIOS call that does console input and stores that location in the patch itself so that the patch can call this routine without having to locate the BIOS every time WordStar wants a character.

In fact, it stores the location of the BIOS call right after a CALL instruction in ORGIN, the patch itself. The expression "$-$" is just a place holder... it evaluates out to 0000H, and will be overwritten by SETPAT before the patch can be called.

When WordStar wants a character, then, it calls ORGIN because we've patched a jump into UCONI. ORGIN, in turn, calls the BIOS just the way WordStar would have. However, when the BIOS call returns with a character it returns to ORGIN, not to WordStar. Having done this we can check to see if the character is a backspace and, if it is, replace it with a rub, 7FH.

Having made the switch, the routine returns to WordStar just as the BIOS call would have done.

Getting It Patched

This patch isn't hardware specific at all, so it will work on virtually any computer which is running CP/M 2.2 and WordStar 3. You'll need a disk with WordStar and its overlay files, ASM.COM and DDT.COM.

To make the patch work, type listing one into a text file called PATCH.ASM using WordStar in the N mode. Get out of WordStar and type

ASM PATCH

If you haven't laid any typos on the file this should assemble with no error messages and return you to CP/M. You'll now have a hex file called PATCH.HEX.

Error messages are characterized by a letter followed by a line of your ASM file being printed on the screen during assembly.

Now type

DDT WS.COM

assuming that your version of WordStar is called WS.COM. You should see something like

```
;               PATCH TO MAKE WORDSTAR THINK IT
;               SEES 7FH DELETE CHARACTER WHEN
;               IT GETS CONTROL H

MORPAT  EQU     02DEH           ;PATCH AREA IN WORDSTAR
UCONI   EQU     02BDH           ;VECTOR TO USER INPUT ROUTINE
INISUB  EQU     02A4H           ;VECTOR TO INITIALIZATION

BS      EQU     'H'-40H         ;BACKSPACE
RUB     EQU     7FH             ;RUB CHARACTER

        ORG     MORPAT

SETPAT  LHLD    1               ;GET POINTER TO BIOS
        LXI     D,6
        DAD     D               ;POINT TO CONIN VECTOR

        SHLD    ORGIN+1         ;SAVE IT
        RET

ORGIN   CALL    $-$             ;CALL THE REAL BIOS VECTOR
        CPI     BS              ;DID IT RETURN A BACKSPACE?
        RNZ                     ;IF NOT, SKIP SUBSTITUTION
        MVI     A,RUB           ;OTHERWISE, MAKE IT RUB CHARACTER
        RET                     ;BACK TO CALLER

        ORG     UCONI
        JMP     ORGIN           ;INSTALL JUMP TO OUR INPUT

        ORG     INISUB
        JMP     SETPAT          ;INSTALL JUMP TO OUR INITIALIZER

        END
```

NEXT PC
4000 0100
—

you should type

IPATCH.HEX
R

and, once the disk has stopped moving, hit control C to get back to the CP/M prompt. Now type

A►SAVE 64 XWS.COM

This will place your patched version of WordStar on the disk under the name XWS.COM. If anything goes wrong you'll still have your original version of WordStar to try again.

Run your new WordStar and try to edit a file. You should be able to backspace over characters using the backspace key just as you can at the CP/M command prompt.

No Leaks

This is a fairly simple patch... you can make more of it if you're up for it. It can, for example, be used to trap the codes emitted by some cursor movement keys and translate them into the cursor diamond control characters WordStar is looking for. You'll need a slightly more elaborate bit of code in ORGIN to do four comparisons. Make sure that your cursor keys aren't already doing characters that WordStar uses for other functions, or you may wind up disablng some of its functions. **CN!**

Will they still be your friends if you say no?

Jerry's a good driver. But this time he's had one too many and the thought of him behind the wheel makes you nervous. Can the gang persuade you to get in the car anyhow or will you stick by your guns and say no?

Nobody who's had too much to drink should ever be on the road. So speak up. Suggest that you or one of the others drive instead. Better still, Jerry should leave the car parked then everyone could share a cab home.

If your friends are really your friends they'll thank you, not put you down, for pointing out the dangers of drinking and driving. What you're really doing is caring about their safety as well as your own. And isn't that what a friend is for?

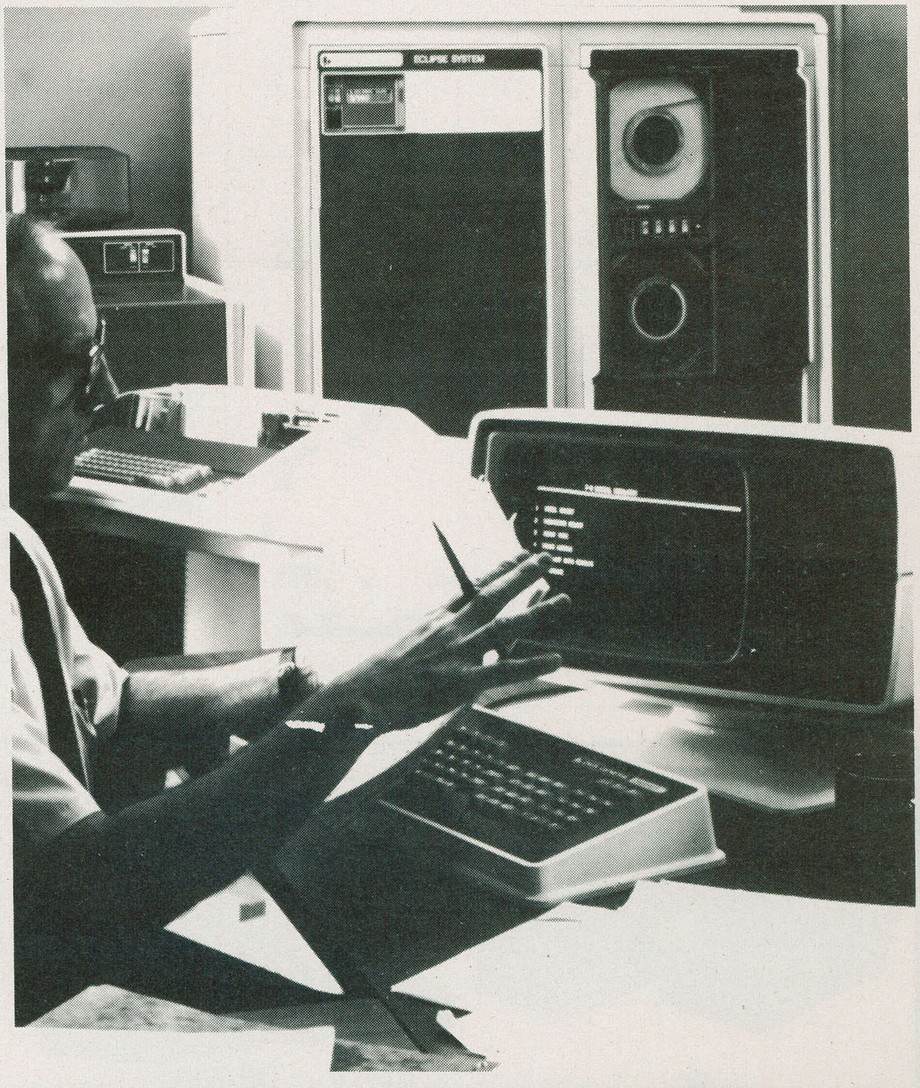
Seagram

We believe in moderation and we've been saying so since 1934.

Do you know how much alcohol you can safely handle? Write to us and we'll send you a valuable free chart on drinking limits. P.O. Box 847, Station H, Montreal, Quebec. H3G 2M8

LIST! Special

Once again we dig into our bag of reader-submitted program listings and present a potful of programs for a number of popular computers to give your fingers a workout and your software budget a break.



Every now and then the second tier of our freelance filing tower sags dangerously and threatens to break through its supporting desk and plummet to the catacombs below. To counteract this gravitational eventuality, we've taken to introducing the occasional special to complement our regular LIST! feature. Computing Now! pays on publication for, and welcomes reader submissions to LIST!, though we ask that potential programmers follow a few guidelines. Debugged program listings should be submitted on white paper, be generated on a computer printer with a reasonably new ribbon, be ideally less than a page in length, have never been published anywhere before and have the author's name and address on the back of the listing... if we use the program, we have to know where to send the cheque.

We cannot print hand-written listings, programs on napkins, typewritten submissions or programs produced in micro-type on calculator plotters.

While the programs appearing in LIST! can't be expected to replace Lotus 1-2-3 in their complexity, they're usually clever and often fun. Too, they can be manipulated for any individual's applications without fear of Lotus' lawyers getting overly excited.

Butterfly Collector
by T. Gray

Catch the butterflies with a pink net controlled by your arrow keys. Not for the timid of heart. Requires a 4K TRS-80 Color Computer.

```
10 REM BUTTERFLY COLLECTOR
20 REM BY T. GRAY
30 REM BOX 39, SUNNYBROOK
40 REM ALBERTA TOC 2M0
50 M=0:BU=0:SC=0:POKE65495,0 'HI
GH SPEED
60 IF PEEK(339)=251 THEN 180
70 CLS(0):C=RND(7)+1
80 X=RND(63):Y=RND(31):A=32:B=16
90 SET(X,Y,C):SET(A,B,4):SC=BU*1
500-INT(M/BU)*10
100 IF X=A AND B=Y THEN FOR T=1
TO 8:SET(X,Y,T):SOUND 20,2:RESET
(X,Y):NEXT:BU=BU+1:SC=BU*1500-IN
T(M/BU)*10):PRINT@0,SC;:GOTO60
110 RESET(X,Y):PRINT@0,SC;
120 IF RND(2)=1 AND X<62 THEN X=
X+1 ELSE IF X>1 THEN X=X-1
130 IF RND(2)=1 AND Y<31 THEN Y=
Y+1 ELSE IF Y>1 THEN Y=Y-1
140 RESET(A,B)
150 IF PEEK(343)=247 AND A>1 THE
N A=A-1:M=M+1 ELSE IF PEEK(344)=
247 AND A<62 THEN A=A+1:M=M+1
160 IF PEEK(341)=247 AND B>1 THE
N B=B-1:M=M+1 ELSE IF PEEK(342)=
247 AND B<31 THEN B=B+1:M=M+1
170 GOTO 90
```

LIST! Special

```
180 CLS:PRINT "YOU CAUGHT ";BU;"
 BUTTERFLIES":PRINT"IN ";M;" MOV
ES FOR A SCORE OF":PRINT SC;" PO
INTS."
```

Piano Animation
by George Dunbar

For those among you who think ZX-81 programs aren't worth peanuts, behold an animated screen of a popular cartoon character hard at work.

```
   5 REM "PIANO ANIMATION"
         BY GEORGE DUNBAR
         WITH APOLOGIES TO
         CHARLES SCHULTZ
  10 PRINT AT 0,15;"  "
  20 PRINT TAB 15;"  "
  30 PRINT TAB 14;"  "
  40 PRINT TAB 14;"  "
  50 PRINT TAB 14;"  "
  60 PRINT TAB 10;"  "
  70 PRINT TAB 7;"  "
  80 PRINT TAB 5;"  "
  90 PRINT TAB 4;"  "
 100 PRINT TAB 3;"  "
 110 PRINT TAB 2;"  "
 120 PRINT TAB 1;"  "
 130 PRINT TAB 1;"  "
 140 PRINT "  "
 150 PRINT "  "
 160 PRINT "  "
 170 PRINT "  "
 180 PRINT "  "
 190 PRINT "  "
 200 PRINT "  "
 210 PRINT "  "
 220 PRINT "  "
 500 PRINT AT 15,16;"  "
 510 PRINT AT 16,16;"  "
 520 PRINT AT 17,16;"  "
```

```
525 PRINT AT 18,16;"▒"
530 PRINT AT 17,15;"▒"
540 PRINT AT 18,15;"▒"
545 PRINT AT 19,15;"▒"
550 PRINT AT 15,16;"▒"
560 PRINT AT 16,16;"▒"
570 PRINT AT 17,16;"▒"
575 PRINT AT 18,16;"▒"
580 PRINT AT 17,15;"▒"
590 PRINT AT 18,15;"▒"
595 PRINT AT 19,15;"▒"
600 GOTO 500
700 SAVE "PIANO ANIMATION"
800 RUN
```

Apple ML Lister

by H.G. Enquist

Without using the Apple's CALL-151 command, this program will format and print to either screen or printer machine language disassemblies. Page length (variable 'L') may be changed in line 90.

```
]LIST
10  REM  ---------------------
20  REM  - M L   L I S T E R -
30  REM  -                   -
40  REM  -  BY H.  ENQUIST   -
50  REM  -  R. R.  1         -
60  REM  -  REDBRIDGE, ON    -
70  REM  -  P0H 2A0          -
80  REM  ---------------------
90 L = 55
100 ML = 768
110  PRINT : INPUT "ENTER START ADDRESS ? "; ST
120  PRINT : INPUT "ENTER END ADDRESS ? "; EN
130 ML = 768
140  FOR J = 0 TO 5: READ D: POKE ML + J, D: NEXT J
150 MS =  INT (ST / 256):LS = ST - (256 * MS)
160  POKE 58, LS: POKE 59, MS
170  PRINT : INPUT "ENTER TITLE ? "; T$
180 PA = 1
190  REM
200  PRINT  CHR$ (12)
210  PRINT : PRINT T$; "   PAGE "; PA
220  PRINT " ": PRINT " "
230  FOR J = 1 TO L
240  CALL ML
250  IF  PEEK (58) + (256 *  PEEK (59)) =  > EN THEN 280
260  NEXT J
270 PA = PA + 1: GOTO 200
280  END
290  DATA  169, 1, 32, 99, 254, 96
```

Apple Painter 1.1

by Edward Mou

Oddly, not everyone has a Koala pad. Create HiRes delights with commands D for plotting, Q for no-trace movement, and using the I,J,K,M,U,O and N keys for positioning.

```
10  REM  APPLE PAINTER 1.1
12  CLEAR : TEXT : HOME : HGR
14 X = 139:Y = 79: HCOLOR= 3
15  HPLOT 0,0 TO 279,0 TO 279,159 TO 0,159 TO 0,0
16  HPLOT X,Y
17  GET Q$
18  IF Q$ = "Q" THEN 44
20  HCOLOR= 3: HPLOT X,Y
22  IF Q$ = "I" THEN Y = Y - 1
24  IF Q$ = "J" THEN X = X - 1
26  IF Q$ = "K" THEN X = X + 1
28  IF Q$ = "M" THEN Y = Y + 1
30  IF Q$ = "U" THEN  GOSUB 78
32  IF Q$ = "O" THEN  GOSUB 80
34  IF Q$ = "N" THEN  GOSUB 82
36  IF Q$ = "," THEN  GOSUB 84
38  HCOLOR= 3: HPLOT X,Y
40  HPLOT X,Y
42  GOTO 17
44  HCOLOR= 3: HPLOT X,Y
46  GET Q$
48  IF Q$ = "D" THEN 40
50  HCOLOR= 0: HPLOT X,Y
52  IF Q$ = "I" THEN Y = Y - 1
54  IF Q$ = "J" THEN X = X - 1
56  IF Q$ = "K" THEN X = X + 1
58  IF Q$ = "M" THEN Y = Y + 1
60  IF Q$ = "U" THEN  GOSUB 78
62  IF Q$ = "O" THEN  GOSUB 80
64  IF Q$ = "N" THEN  GOSUB 82
66  IF Q$ = "," THEN  GOSUB 84
68  HCOLOR= 3
70  HPLOT X,Y
72  HCOLOR= 0
74  HPLOT X,Y
76  GOTO 44
78 X = X - 1:Y = Y - 1: RETURN
80 Y = Y - 1:X = X + 1: RETURN
82 X = X - 1:Y = Y + 1: RETURN
84 Y = Y + 1:X = X + 1: RETURN
```

Disk Formatter

by Ivan Williams

Vic 20 users with disk drives have never had it so easy. A quick utility to format as many disks as you need in one sitting.

```
1 REM THIS UTILLITY PROGRAM HELPS
2 REM YOU TO FORMAT DISKS
3 REM FOR THE VIC-20  ONLY!
4 REM BY IVAN WILLIAMS
5 PRINT"  DISK FORMATTER  ":PRINTCHR$(8)
6 PRINT"INSERT DISK....!"
7 PRINTSPC(9)"AND":PRINT"PRESS  RETURN "
8 GETA$:IFA$=""THEN8
9 IFA$=CHR$(13)THEN12
10 IFA$<>CHR$(13)THEN8
12 PRINT"FORMATTING WILL START":PRINT"IN 15
   SECONDS...!"
14 TI$="000000"
15 FORT=1TO2000:NEXT
16 OPEN15,8,15:PRINT#15,"N:DISK NAME,ID":CLOSE15
17 IFST=1THEN25
18 PRINT"YOUR DISK IS NOW":PRINT" READY "
19 PRINT"ARE YOU DONE?[Y/N]"
20 GETA$:IFA$<>"Y"ANDA$<>"N"THEN20
22 IFA$="Y"THEN24
23 IFA$="N"THENGOTO5
24 CLR:END
25 PRINT"ERROR:YOU HAVE A      PROBLEM WITH YOUR DISK":STOP
```

Vic Strobe

by Tony Savor

Remember disco? Remember Flashdance? Bring back those memories of happy feet with a strobe light for the Vic 20. Not to be taken internally.

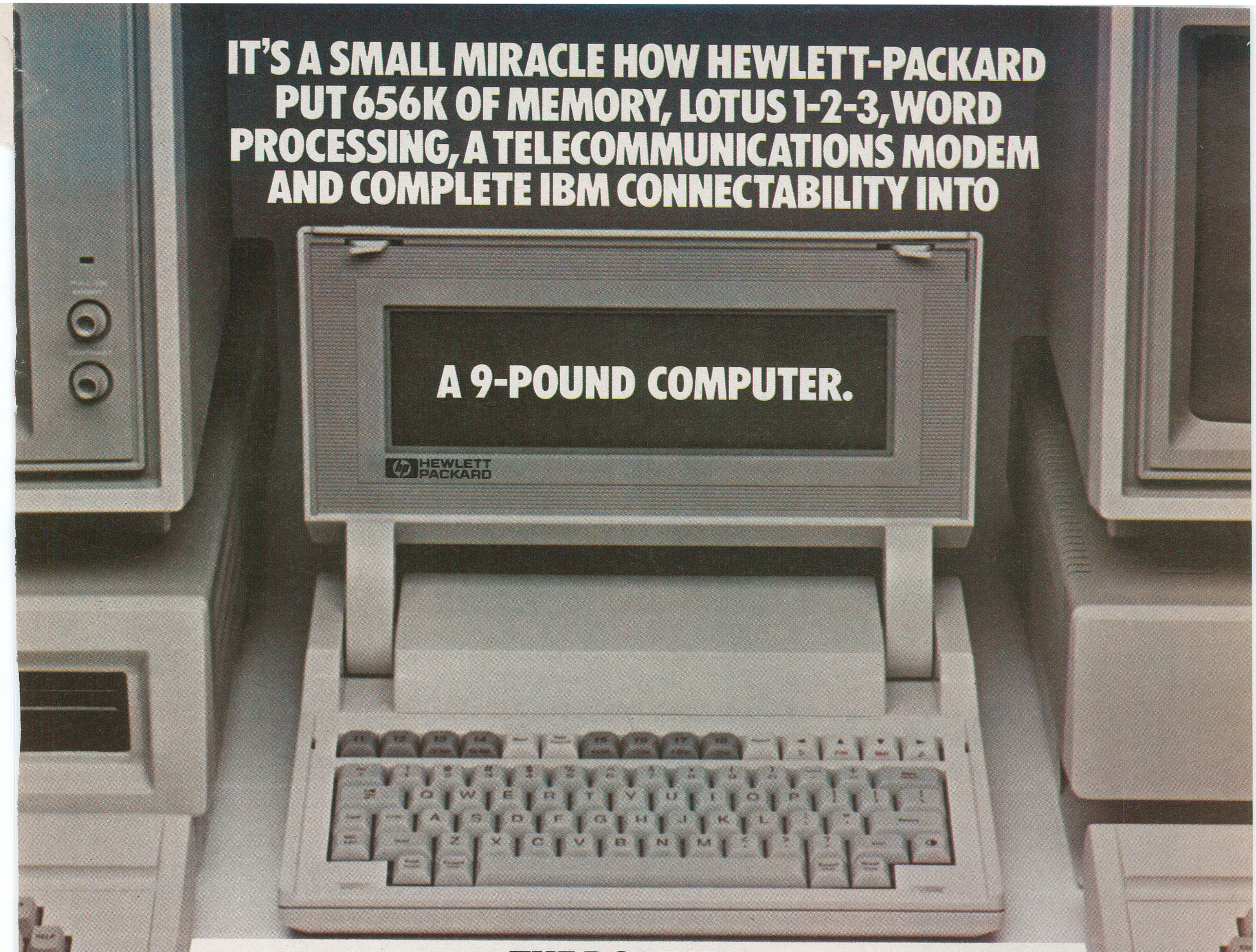
```
1 REM BY TONY SAVOR         1998 GLENDALE DR      PICKERING ONTARIO
2 REM VIC STROBE,            CONVERTS THE           VIC-20 INTO A          STROB
E LIGHT
5 REM TO TYPE IN THE        INSTRUCTIONS IN       THE SQUARE             BRACKETS
 TYPE
6 REM CURSOR MOVEMENT       THEY REPRESENT.       FOR EXAMPLE IF
7 REM YOU SEE [2 DWN]       THEN HIT THE
8 REM CURSOR UP/DOWN        KEY TWICE, OR IF      YOU SEE [CLR]
9 REM HIT THE SHIFT         AND CLR KEY.
10 REM IF YOU SEE            [RVS] THEN HIT        CONTROL AND 9.
11 REM FOR [F1] JUST         HIT THE F1 KEY.
40 PRINT"[CLR][CSR DWN][RVS]VIC STROBE[OFF]":PRINT"[2 DWN]BY TONY SAVOR":PRINT"-
----------------"
41 PRINT"[DWN]WHEN RUNNING HIT F1      TO CHANGE SPEED"
42 PRINT"    AND HIT F7 TO QUIT"
50 PRINT"[3 DWN]SPEED (1-500)":PRINT"[2 RIGHT]460[5 LEFT][UP]":INPUTT
51 IFT<1ORT>500THENRUN
52 T=500-T
59 PRINT"[CLR]"
60 POKE36879,25:FORX=1TOT:NEXT
70 POKE36879,8:FORX=1TOT:NEXT
80 GETA$:IFA$="[F1]"THENPOKE36879,27:RUN
81 IFA$="[F7]"THENSYS50010
90 GOTO60
```

LIST! Special

HiRes Printer Utility
by Neal A. Bridges

Despite rumours to the contrary, the Gemini 10X and the Commodore 64 can get along together. This utility will dump C64 HiRes screens from memory to the printer.

```
0 rem c64/gemini 10x hires printer utility
1 rem ****************
2 rem * written by   *
3 rem * neal bridges *
4 rem ****************
5 :
6 iff=1then1020
60 poke51,0:poke52,32:poke56,32:clr:printchr$(14):poke53280,15:poke53281,1
120 rem menu
180 print"{CLR}{C/DN}{C/RT}{C/RT}{RVON}Main Menu"
240 print"{C/DN}{C/RT}1)  Load a picture from disk."
300 print"{C/DN}{C/RT}2)  Print the picture in memory."
360 print"{C/DN}{C/RT}3)  View the picture in memory."
420 print"{C/DN}{C/RT}4)  Reverse the picture in memory."
480 print"{C/DN}{C/RT}5)  QUIT the program."
540 geta$:ifa$=""then540
600 ifa$<"1"ora$>"5"then540
660 a=val(a$):onagoto720,1140,2940,2760,3480
720 rem load a picture from disk
780 print"{CLR}{C/DN}{C/RT}Enter the filename";:inputf$
840 iff$=""then120
900 print"{CLR}{C/DN}{C/RT}Loading...  Please wait."
960 iff=0thenf=1:loadf$,8,1
1020 f=0
1080 goto120
1140 print"{CLR}{C/DN}{C/RT}{C/RT}{RVON}Printing Menu"
1200 print"{C/DN}{C/RT}1)  Set the left-hand margin."
1260 print"{C/DN}{C/RT}2)  Print the picture on the printer."
1320 print"{C/DN}{C/RT}3)  Return to the main menu."
1380 geta$:ifa$<"1"ora$>"3"then1380
1440 a=val(a$):onagoto1500,2100,120
1500 print"{CLR}{C/DN}{C/RT}Remember, the hi-res picture will"
1560 print"{C/DN}{C/RT}be printed on its side, with the top"
1620 print"{C/DN}{C/RT}facing the right-hand edge of the"
1680 print"{C/DN}{C/RT}page."
1740 print"{C/DN}{C/RT}Enter x to return to the menu."
1800 print"{C/DN}{C/RT}Enter the margin value (0-40)";:inputa$
1860 ifa$="x"then1140
1920 t=val(a$)
1980 ift<0ort>40then1500
2040 goto1140
2100 print"{CLR}{C/DN}{C/RT}Is the printer ready?"
2160 geta$:ifa$=""then2160
2220 ifa$="n"then1140
2280 ifa$<>"y"then2160
2340 print"{CLR}{C/DN}{C/RT}OK, I'm sending the picture to the"
2400 print"{C/DN}{C/RT}printer.  Please wait..."
2460 open4,4,5:print#4,chr$(27)"3"chr$(16);chr$(27)"m"chr$(t);
     :e=8192:h=320:r=8
2520 fori=.to39:forj=24to.step-1:fork=7to.step-1:b$=b$+chr$
     (peek(e+h*j+r*i+k))
2580 nextk,j:print#4,chr$(27)"k"chr$(200)chr$(0)b$
2590 b$=""
2600 nexti
2640 print#4,chr$(27)"@";:print#4:close4
2700 goto1140
2760 print"{CLR}{C/DN}{C/RT}Are you sure";:inputa$:ifa$<>"y"
     then120
2820 print"{C/DN}{C/RT}Please wait... this takes a little
     {C/DN}while."
2880 fori=8192to16384:pokei,255-peek(i):next:goto120
2940 print"{CLR}{C/DN}{C/RT}Press any key to view,"
3000 print"{C/DN}{C/RT}press it again to quit."
3060 geta$:ifa$=""then3060
3120 print"{CLR}"
3180 fori=1024to2024:pokei,1:next
3240 poke53272,peek(53272)or8
3300 poke53265,peek(53265)or32
3360 geta$:ifa$=""then3360
3420 poke53272,23:poke53265,27:goto120
3480 print"{CLR}{C/DN}{C/RT}Goodbye...":end
3600 nexti
```



THE PORTABLE.

The Portable is designed with more total memory than most leading desktop personal computers...656K, in fact. That includes 272K of user memory. So, The Portable's built-in business software can work with enormous amounts of data.

1-2-3™ from Lotus™ is permanently built into The Portable. So is Hewlett-Packard's word processing program, MemoMaker. Just press the key and you're ready to work.

The Portable even has a built-in modem and easy-to-use telecommunications software to send or receive data using a standard telephone jack.

The Portable's rechargeable battery gives you 16 hours of continuous use on a single charge.

And finally, you can work comfortably on a full-size keyboard and an easy-to-read 16-line by 80-column screen.

And all of this folds shut to turn The Portable into a simple nine-pound box.

For a personal demonstration and the Hewlett-Packard dealer nearest you, call 1-800-387-3867.

1-2-3 and Lotus are trademarks of Lotus Development Corporation.

Circle No. 2 on Reader Service Card

HEWLETT PACKARD

6877 Goreway Drive
Mississauga, Ontario
L4V 1M8

This coupon worth up to $250.00

Complete this coupon and take it to your participating HP Dealer for a personal demonstration of The Portable and your choice of the following when you purchase The Portable from now until February 28, 1985.

- A free interface card for the IBM® PC
- A free interface card for the HP Touchscreen PC
- $200.00 off these HP products: Touchscreen PC, ThinkJet or LaserJet Printers, Plotters and Disc Drives.

IBM is a registered trademark of International Business Machines Corporation.

Name________________________

Title (if applicable)____________________

Company (if applicable)____________________

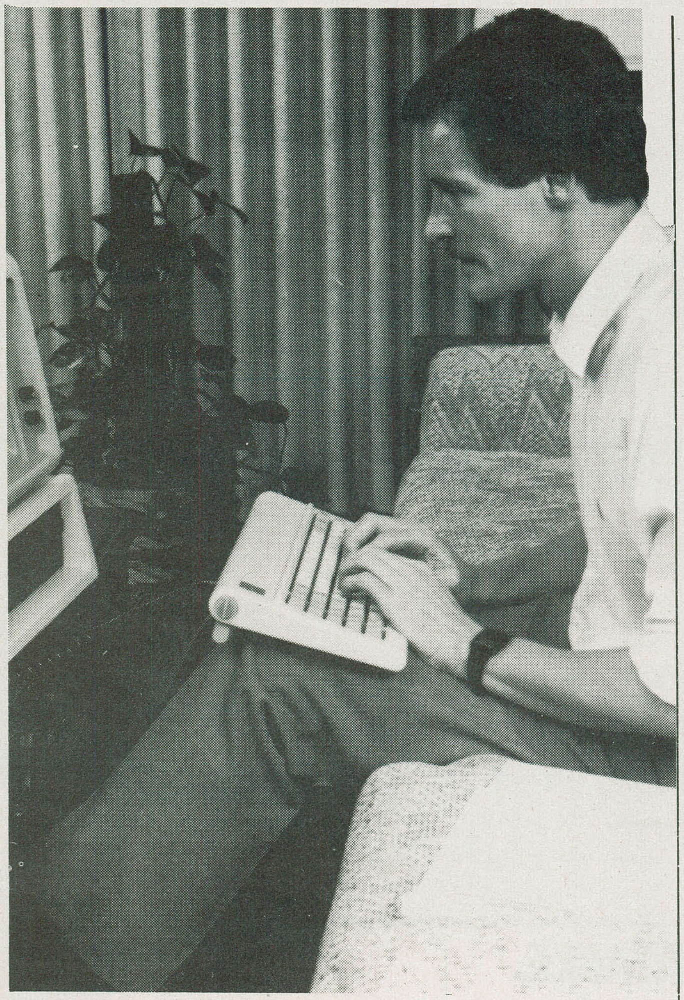
Telephone (   )____________________

Street________________________

City/Town________________________

Province____________ Postal Code____________

Dem PC Utility Blues



Writing small assembler programs for the IBM PC takes a couple of tricks they don't seem to see fit to discuss in the manuals. Here's a look at how to tickle the assembler.

by Steve Rimmer

Some computers just seem to arise in the small hours before dawn, crawl up on the battlements and howl mournfully at the moon for little programs to run. The IBM PC is certainly one of these. While writing bits of code for it may seem to be a lot more daunting than would be the case for, say, the Apple, it has a plethora of holes to fill and functions to call.

The structure of MS-DOS... yes, there is one in there, right beside the random error message generator... is such that utility programs and other quick commands can be handled quite flexibly. One can, for example, set up a special utility directory on one's disk, create a permanent path into it and use the resulting redirection to add illusionary built in commands to the system.

The commands that come with the PC are useful in their own way, but one needs only trespass a few paces beyond the safety of the blue pasture to find that there are demons out there that aren't in the least bit impressed with what came with your system master. They're big, slavering demons, too, hairy brutes with lots of eyes and feet that haven't been washed since the last coming of the swamp king.

You're not going to blast *them* into oblivion with a few lines of BASIC, me lads. It's gonna take an assembler.

Romancing the Phone

The writing of code to put down demons and make one's computer a safe and cheerful abode once more isn't as awesome a task as it might seem. There is a seldom used function of the number twenty-one interrupt which neutralizes spells and curses... set the DX register to point to the spell or curse in question.

More conventional applications, however, require technique and other nasty academic things. While one can write assembler programs for the PC with fairly little cerebral overhead, a few tricks will make the experience a lot more real.

The first juggling act one should be aware of in writing assembly language programs for the IBM is in the difference between EXE and COM files. A COM file is just a block of code. When the system gets one it will load it into the current segment, beginning at 0100H, and run it. May the creator of all life and sentient dashboards help you if it wasn't written to run there.

An EXE file, on the other claw, has a six hundred or so byte header which contains all sorts of manifest information, including an address which specifies where the file is supposed to be when it runs. As such, an EXE file can be specified to load anywhere at all.

Writing a program as an EXE file is a really good trip for some of the more sophisticated things one might do on the PC... it's essential, for example, if you anticipate writing a program that will result in more than sixty four kilobytes of code.

The programs we'll be looking at here have about sixty three and a half K of overhead... you can relax.

In fact, EXE files are wasteful of space and access time when you're planning to write simple programs. What's more, you have to do a lot more finessing with your code to make them happen, kissing the feet of DOS every time one boots to make the nasty thing return properly.

While the writing of programs which turn out to be COM files is poorly explained in most manuals, it's actually much simpler than doing EXE's. The only drag is that it takes one additional step through each assembly.

To BIN or Not To BIN

To begin with, if you've checked out some of the earlier assembler programs we've run in Computing Now! for the PC,

Listing 1, The RTTY Program

```
;       ......................................
;      .                                      .
;      .        RADIO TELETYPE DECODER        .
;      .          & karmic footbath           .
;      .         copyright (c) 1984           .
;      .            Steve Rimmer              .
;      .                                      .
;       ......................................
;
;     This little fruit bat takes BAUDOT code fed
;     into the COM1 serial port and makes it into
;     ASCII it can display on the tube, which it
;     does. BAUDOT is not easily translated, as it
;     consists of two sets of thirty two characters,
;     or cases. The active case is the one most
```

Almost Free PC Software Volume II

It must have been the roses

A good program is like a good politician . . . no, wait, we've succeeded in finding some good programs. However, it did take a lot of searching. Presented here is a selection of some of the best utilities, games, programmers' tools and business applications ever to order the bytes on a disk.

Sweep is a turbocharged Ferrarri of a disk utility which makes the COPY command look like a goat herd by comparison. It allows one to do mass copying, deletion, renaming and other disk functions all in menu driven comfort. It supports essentially the same command structure and behavior as the CP/M Sweep and Disk programs.

Worldmap is a sophisticated graphics program which draws a very detailed picture of the planet we live on and daily endeavour to blow up. It will display its wares on the tube or send them out to a printer.

Anitra plays Anitra's Dance by Edvard Grieg. PC music programs are a gas . . . everyone should have a disk full of them.

Ramdisk is among the most useful of all the utilities you'll ever plug into your PC. It creates a virtual drive on your system out of memory. You can pop your files over to it when you boot the beast and thereafter experience disk accesses that take less time to complete than real drives take to turn on their LEDs.

Alien plays a bizarre adventure game. It leads you into some pretty warped places. It comes with a massive data file for an adventure that you won't get tired of 'til the dragons come home for the evening.

FOS is a personal financial manager which will, among other things, make your cheque books into servants of humanity as opposed to denizens of the aforementioned adventure game. It's thunderously slick.

Jukebox represents yet another PC music system. This one comes with a host of songs to play and some really electric graphics.

Asmgen is one of the best text disassemblers we've come across. It takes any executable COM or EXE file and produces an assembler listing. It's surprisingly good at distinguishing between code and imbedded data or text. If you have need to patch or modify code this thing will outdo DEBUG by light years.

Struct will appeal to the rabid programmer in everyone. It allows MASM to be used to assemble a sort of higher level language. Included also is a test file to illustrate the syntax.

Prtsc replaces the internal PC screen dump code with something more suited to reality. It allows one to hit the PrtSc* key and then select what the screen dump will look like from a menu. It supports a number of popular printers.

Breakout plays a PC version of the popular game. It will accept input from either a joystick or the keyboard. The graphics are good and the action is adjustable from a beginner's level right up to fast and nasty.

Util is a collection of system utilities all under one menu driven roof. Among its many talents are a sorted directory, keyboard redefinition and the facility for scrolling up and down through a text file.

All of this software is available on a single disk. It comes with extensive on disk documentation to explain how to make it do its things. The whole works cost a mere

$19.95

Almost Free PC Software Volume II
Computing Now! Magazine
25 Overlea Boulevard, Suite 601
Toronto, Ontario M4H 1B1
Ontario residents please add 7% sales tax.

Or Order it by phone on your Visa, Mastercard or American Express. Call 1-423-3262 during regular (EST.) business hours.

Fine Print: This software is all in the public domain. It is offered here without charge. The price of this package defers our cost in collecting and preparing it, the cost of the media, postage and handling.

While we have made every effort to ensure that this software is functional and will do what it's supposed to, we are unable to offer technical assistance in adapting it for your application.

Dem PC Utility Blues

```
;           recently selected by one of two case characters.
;           As such, the conversion process uses two
;           translation tables.

;           Quite a lot more about BAUDOT can be found in
;           Computing Now! for March 1984.

LF          EQU     10          ;LINE FEED
CR          EQU     13          ;CARRIAGE RETURN

CODEX       SEGMENT
MAIN        PROC    FAR
            ASSUME  CS:CODEX
            ORG     0100H

START:      CALL    INITMOD
            CALL    CLS
            CALL    ILPRT
            DB      '[ Wombat Brothers Radio Teletype Decoder '
            DB      'and Karmic Footbath ]',CR,LF
            DB      '[ Copyright 1984 (c) Rimmer ]',CR,LF
            DB      '[ Awaiting stuff... hit any key to be gone ]'
            DB      CR,LF,LF,0

TERLP:      MOV     AH,1
            INT     16H                 ;SEE KEYBOARD STATUS
            JNZ     EXIT                ;IF CHARACTER, BE GONE
            MOV     DX,3FDH             ;POINT TO MODEM STATUS
            IN      AL,DX               ;GET THE MODEM STATUS
            TEST    AL,00000001B        ;IF NO CHAR WAITING...
            JZ      TERLP               ;KEEP LOOPING

            MOV     DX,3F8H             ;GET CHARACTER FROM MODEM
            IN      AL,DX               ;
            AND     AL,1FH              ;MASK PARITY
            CALL    CONVERT             ;MAKE IT INTO ASCII
            CMP     AL,0                ;SEE IF IT'S CASE CHANGE
            JZ      TERLP               ;...NO DISPLAY IF IT IS
            MOV     BX,0
            MOV     AH,14
            INT     10H                 ;OUTPUT IT TO TUBE
            JMP     TERLP               ;GET ANOTHER CHARACTER
;
EXIT:       CALL    ILPRT               ;SAY BYE BYE
            DB      CR,LF,LF
            DB      '[ Off line... Returning to DOS ]',CR,LF,0
            INT     20H                 ;AND SCOOT

;           TABLES TO TRANSLATE FROM BAUDOT TO ASCII
;           IT LOOKS AT EVERY ODD ONE, REPLACING AL WITH
;           THE PREVIOUS EVEN ONE IF IT MATCHES, OR A
;           GRINNING FACE IF NOTHING MATCHES

LTABL:      DB      'A',03H,'B',19H,'C',0EH,'D',09H,'E',01H,'F',0DH
            DB      'G',1AH,'H',14H,'I',06H,'J',0BH,'K',0FH,'L',12H
            DB      'M',1CH,'N',0CH,'O',18H,'P',16H,'Q',17H,'R',0AH
            DB      'S',05H,'T',10H,'U',07H,'V',1EH,'W',13H,'X',1DH
            DB      'Y',15H,'Z',11H,' ',04H
            DB      0DH,08H,0AH,02H,1BH,1BH,1FH,1FH,00H,00H

CTABL:      DB      '-',03H,'7',19H,':',0EH,'$',09H,'3',01H,21H,0DH
            DB      '&',1AH,'#',14H,'8',06H,27H,0BH,'(',0FH,')',12H
            DB      '.',1CH,',',0CH,'9',18H,'0',16H,'1',17H,'4',0AH
            DB      07H,05H,'5',10H,'7',07H,3BH,1EH,'2',13H,'/',1DH
            DB      '6',15H,'"',11H,' ',04H
            DB      0DH,08H,0AH,02H,1BH,1BH,1FH,1FH,00H,00H

CASE:       DB      1FH                 ;CASE FLAG... STARTS IN LETTERS
```

```
MAIN    ENDP

CONVERT PROC    NEAR            ;COVERT BAUDOT TO ASCII
        CMP     AL,1BH          ;IS IT FIGURES SET CHARACTER?
        JZ      SHIFT           ;IF SO, STASH IT AND BOOGIE
        CMP     AL,1FH          ;IS IT LETTERS SET CHARACTER?
        JZ      SHIFT           ;IF SO, LIKEWISE AND THE SAME
        MOV     BX,OFFSET CASE  ;POINT TO CASE FLAG
        MOV     AH,[BX]         ;GET MOST RECENT CASE SETTING
        CMP     AH,1FH          ;SEE IF IT'S A LETTERS FLAG
        JZ      LETR            ;AND GO TO LETTER ROUTINE IF SO

FIGR:   MOV     BX,OFFSET CTABL+1       ;POINT TO TABLE
        MOV     CX,32           ;32 CHARACTERS IN BAUDOT CASE
FIGL:   MOV     AH,[BX]         ;GET TABLE ENTRY
        CMP     AH,AL           ;SEE IF IT MATCHES CHARACTER
        JZ      MATCH           ;IF SO, DO SWITCH
        INC     BX              ;OTHERWISE...
        INC     BX              ; ...POINT INTO NEXT ENTRY
        LOOP    FIGL            ;AND GO AGAIN
        JMP     NOCON           ;IF NO MATCH, SHOW FACE CHARACTER

LETR:   MOV     BX,OFFSET LTABL+1       ;POINT TO TABLE
        MOV     CX,32           ;32 CHARACTERS IN BAUDOT CASE
LETL:   MOV     AH,[BX]         ;GET TABLE ENTRY
        CMP     AH,AL           ;SEE IF IT MATCHES CHARACTER
        JZ      MATCH           ;IF SO, DO MATCH
        INC     BX              ;OTHERWISE
        INC     BX              ; ...POINT INTO NEXT ENTRY
        LOOP    LETL            ;AND GO AGAIN
        JMP     NOCON           ;IF NO MATCH, SHOW FACE CHARACTER

SHIFT:  MOV     BX,OFFSET CASE  ;POINT AT CASE FLAG
        MOV     [BX],AL         ;MOVE NEW CASE INTO IT
        MOV     AL,0            ;SET AL TO ZERO FOR NO DISPLAY
        JMP     CN_RET          ;AND RETURN

NOCON:  MOV     AL,1            ;MOVE SMILING FACE INTO AL
        JMP     CN_RET          ;AND GO SHOW IT

MATCH:  DEC     BX              ;MOVE BACK ONE TABLE BYTE
        MOV     AL,[BX]         ;GET ASCII CHARACTER

CN_RET: RET                     ;RETURN TO MAIN LOOP

CONVERT ENDP

POSIT   PROC    NEAR
;SET CURSOR POSITION IN DX
        MOV     AH,15
        INT     10H             ;GET DISPLAY PAGE
        MOV     AH,2
        INT     10H             ;SET CURSOR POSITION IN DX
POSIT   ENDP

INITMOD PROC    NEAR
        MOV     DX,3FBH
        MOV     AL,80H          ;OPEN DLAB
        OUT     DX,AL
        MOV     DX,3F8H
        MOV     AL,0E6H         ;SET LOW ORDER DIVISOR
        OUT     DX,AL
        MOV     DX,3F9H
        MOV     AL,09H          ;SET HIGH ORDER DIVISOR
        OUT     DX,AL
        MOV     DX,3FBH
CFW:    MOV     AL,04H          ;SET CFW, 5 BITS 1.5 STOPS
        OUT     DX,AL
        MOV     DX,3FCH
        MOV     AL,00000011B    ;SET LOCAL LOOPBACK OFF
        RET
INITMOD ENDP

CLS     PROC    NEAR            ;CLEAR THE SCREEN
        MOV     CX,0
        MOV     BH,7
        MOV     DH,24
        MOV     DL,79
        MOV     AL,0
        MOV     AH,6
        INT     10H
        MOV     DX,0
        CALL    POSIT           ;HOME CURSOR
        RET
CLS     ENDP

ILPRT   PROC    NEAR
;IN LINE PRINT
        POP     BX              ;GET LOCATION OF FIRST BYTE
ILPLP:  MOV     DL,[BX]         ;AND GET BYTE INTO DL
        CMP     DL,0            ;IF NULL, WE'RE DONE
        JE      ILPRET          ; ...SO GO HOME
        MOV     AH,2            ; ...OTHERWISE, SHOW CHARACTER
        PUSH    BX              ;SAVE POINTER FIRST
        INT     21H
        POP     BX              ;GET POINTER BACK
        INC     BX              ;POINT TO NEXT BYTE
        JMP     ILPLP           ;AND LOOP
ILPRET: INC     BX              ;POINT TO INSTRUCTION AFTER NULL
        PUSH    BX              ;AND PUSH ADDRESS ON STACK
        RET                     ;RETURN
ILPRT   ENDP

CODEX   ENDS
        END     START
```

you'll recall all the heavy manipulations which happened at the beginning of each chunk of code. This field of ASSUME's and PUSH's is a standard prolog to keep the operating system smiling. The equivalent bowing and scraping for a prospective COM file is a lot less intense.

```
CODEX   SEGMENT
MAIN    PROC FAR
ASSUME  CS:CODEX
ORG     0100H
```

and then, when the whole cotillion has ended,

```
INT     20H
MAIN    ENDP
CODEX   ENDS
```

which returns everything smartly to DOS... probably ready to throw some more random errors.

You'll note that COM files, unlike EXE's, use the ORG pseudo-op. This defines that the code *has* to start at 0100H. In the case of an EXE program, the assembler figures out where the code segment is going to be and, as such, sets the ORG by itself.

Using an ORG of other than 0100H will produce all manner of colourful results.

The other aspect of writing programs which will turn out as COM files is that they are assembled slightly differently. Taking the case of the BAUD program in this article... which would presumably start its life as BAUD.ASM, one would say.

```
MASM BAUD
```

hit some carriage returns to fox the additional files we aren't going to use anyway, and then

```
LINK BAUD
```

with some more carriage returns, and finally

```
EXE2BIN BAUD BAUD.COM
```

which converts the EXE file to a COM file.

This is as tedious as a car salesman on a muggy Wednesday afternoon. It involves a lot of typing, and is so un-high tech as to be blasphemous. There is a much better way. You'll want to create an assembler batch file early on. Do the following stuff.

```
A>COPY CON: ASM.BAT
MASM /1 %1 NUL NUL
```

Dem PC Utility Blues

Listing 2, The BAUD Program

```
; ......................................
;
;       BAUD RATE SETUP UTILITY FOR
;       IBM PC COM1 PORT
;
;           Copyright (c) 1984
;              Steve Rimmer
;
;       this code likes cheese quite a lot
;
; ......................................
;
CR      EQU     13
LF      EQU     10
CMDTL   EQU     0082H               ;LOCATION OF COMMAND TAIL

CODEX   SEGMENT
MAIN    PROC    FAR
        ASSUME  CS:CODEX
        ORG     0100H
START:  JMP     OVER

BD50:   DW      0009H
BD75:   DW      0006H
BD110:  DW      1704H
BD134:  DW      5903H
BD150:  DW      0030H
BD300:  DW      8001H
BD600:  DW      0C000H
BD1200: DW      6000H
BD1800: DW      4000H
BD2000: DW      3A00H
BD2400: DW      3000H
BD3600: DW      2000H
BD4800: DW      1800H
BD7200: DW      1000H
BD9600: DW      0C00H

MSG1:   DB      CR,LF,'Wombat Brothers Baud Rate Setup Utility'
        DB      CR,LF,'   Copyright 1984 (c)  Steve Rimmer'
        DB      CR,LF
        DB      CR,LF,'     The available baud rates are'
        DB      CR,LF,'     50          75          110'
        DB      CR,LF,'     134.5       150         300'
        DB      CR,LF,'     600         1200        1800'
        DB      CR,LF,'     2000        2400        3600'
        DB      CR,LF,'     4800        7200        9600'
        DB      CR,LF
        DB      CR,LF,'Normal use:'
        DB      CR,LF,'A>BAUD 1200 [choose any baud rate]'
        DB      CR,LF
        DB      CR,LF,'This program is not for use by neo-gothic'
        DB      CR,LF,'Northern Latvian dwarfs of any age or sex.'
        DB      CR,LF
        DB      CR,LF
        DB      CR,LF,'$'
;
;
;FIRST OFF, CHECK TO SEE IF THERE'S A VALID PARAMETER
OVER:
        MOV     DX,CMDTL
        MOV     CX,OFFSET BD50
        CALL    STRCOMP
        DB      '50',0
        JC      OVER1
        JMP     SETBD
OVER1:  INC     CX
        INC     CX
        CALL    STRCOMP
        DB      '75',0
        JC      OVER2
        JMP     SETBD
OVER2:  INC     CX
        INC     CX
        CALL    STRCOMP
        DB      '110',0
        JC      OVER3
        JMP     SETBD
OVER3:  INC     CX
        INC     CX
        CALL    STRCOMP
        DB      '134.5',0
        JC      OVER4
        JMP     SETBD
OVER4:  INC     CX
        INC     CX
        CALL    STRCOMP
        DB      '150',0
        JC      OVER5
        JMP     SETBD
OVER5:  INC     CX
        INC     CX
        CALL    STRCOMP
        DB      '300',0
        JC      OVER6
        JMP     SETBD
OVER6:  INC     CX
        INC     CX
        CALL    STRCOMP
        DB      '600',0
        JC      OVER7
        JMP     SETBD
OVER7:  INC     CX
        INC     CX
        CALL    STRCOMP
        DB      '1200',0
        JC      OVER8
        JMP     SETBD
OVER8:  INC     CX
        INC     CX
        CALL    STRCOMP
        DB      '1800',0
        JC      OVER9
        JMP     SETBD
OVER9:  INC     CX
        INC     CX
        CALL    STRCOMP
        DB      '2000',0
        JC      OVER10
        JMP     SETBD
OVER10: INC     CX
        INC     CX
        CALL    STRCOMP
        DB      '2400',0
        JC      OVER11
        JMP     SETBD
OVER11: INC     CX
        INC     CX
        CALL    STRCOMP
        DB      '3600',0
        JC      OVER12
        JMP     SETBD
OVER12: INC     CX
        INC     CX
        CALL    STRCOMP
        DB      '4800',0
        JC      OVER13
        JMP     SETBD
OVER13: INC     CX
        INC     CX
        CALL    STRCOMP
        DB      '7200',0
        JC      OVER14
        JMP     SETBD
OVER14: INC     CX
        INC     CX
        CALL    STRCOMP
        DB      '9600',0
        JC      NOPARM
;FALL THROUGH TO BAUD RATE SETUP

;WE HAVE A VALID BAUD RATE PARAMETER
SETBD:  MOV     BX,CX
        MOV     AX,[BX]
        PUSH    AX
        MOV     DX,3FBH
        MOV     AL,80H
        OUT     DX,AL
        POP     AX
        MOV     DX,3F9H
BDLSB:  OUT     DX,AL               ;SET LOW ORDER DIV
        MOV     DX,3F8H
        MOV     AL,AH
BDMSB:  OUT     DX,AL               ;SET HIGH ORDER DIV
        MOV     DX,3FBH
CFW:    MOV     AL,1AH    ;SET CFW
        OUT     DX,AL
        MOV     DX,3FCH
        MOV     AL,00000011B        ;SET LOCAL LOOPBACK
        JMP     EXIT
```

```
; if there's no parameter, then show help message
NOPARM: MOV     DX,OFFSET MSG1
        MOV     AH,9
        INT     21H
;
EXIT:   INT     20H                     ;BACK TO DO
MAIN    ENDP
;
;
;       +++ SUBMARINES +++
;
;
STRCOMP PROC    NEAR
;IN LINE STRING COMPARE
;STRING TO COMPARE TO IN D, OTHER STRING IN LINE
;RETURNS WITH CARRY CLEAR IF EQUAL
        POP     BX
        PUSH    DX
        PUSH    CX
        PUSH    AX
        MOV     CL,0                    ;CLEAR FLAG
LLP:    MOV     AH,[BX]
        CMP     AH,0
        JE      DCOM
        PUSH    BX
        MOV     BX,DX
        MOV     AL,[BX]
        POP     BX
        CMP     AH,AL
        JE      NOFLAG
        MOV     CL,OFFH                 ;SET FLAG
NOFLAG: INC     DX
        INC     BX
        JMP     LLP
DCOM:   CLC
        CMP     CL,0
        JE      NOSET
        STC
NOSET:  POP     AX
        POP     CX
        POP     DX
        INC     BX
        PUSH    BX
        RET
STRCOMP ENDP

CODEX   ENDS
;
END     START
```

```
LINK %1 @AUTOLINK
EXE2BIN %1 %1.COM
DEL %1.OBJ
```

and then hit control Z. You will also need a file called AUTOLINK which contains four carriage returns.

Once you've got all this happening, just type

ASM BAUD

and the file will do the rest for you. Pick up your trusty guitar, nose harp or other sonic disturbance, crack open a cold penguin and watch the circus.

There is one minor glitch in this system. The linker will lay a lack of stack error on you. Ignore this... it's just in a bad mood.

Now, The Code

The two programs which accompany this feature are representative of the sorts of things people write little COM files to do. While you might not have need for these particular programs, you can warp their bits around and come up with other things.

The baud program is quite useful It sets up the baud rate and the other communications parameters.

The function of BAUD.COM... when you finally get it assembled... is to take an argument from the command line and set the baud rate accordingly. That's a five minute argument, as opposed to the full half hour.

The usual form of using this thing is

BAUD 300

which you'll already know about if you've looked at the code, as it lays a help message on you if you don't give it a parameter, or give it a wrong one.

There are couple of useful things happening in BAUD. The first is in finding out how the command line works.

When you boot a COM file MS-DOS does a number of preparatory things for you. The most useful of these... at least from the point of view of this program... is its propensity for placing the command line prompt at a fairly predictable location, to wit, 0082H in whatever segment the program finds itself. It's a lot like CP/M in this respect.

Knowing where this string is to be found, it can be compared to fixed strings to see what the program has been told to do.

This program features one of two very useful subroutines. This one is called STRCOMP, and it behaves fairly strangely. The form of calling this thing is

```
MOV     DX,STRING
   or
MOV     DX,OFFSET STRING

CALL    STRCOMP
DB      'ELECTRIC WOMBATS',0
```

followed by the rest of the code. It will compare the string pointed to by DX with the one in the DB after it. The string in the DB must be terminated by a null, or it will compare one half of the computer with the other half.

The routine returns with the carry flag clear if the strings are equal.

Now, this is all very convenient, because you don't have to go screaming up to the top of your file every time you want to put a string somewhere to compare with. All the strings are in line. However, if you are following what's happening here you will realize that when STRCOMP returns the program will try to execute the string and very likely crash in the attempt.

In fact, this doesn't happen because STRCOMP fiddles the stack. When you call STRCOMP the 8088 pushes the return address for the call onto the stack, this being the address of what it thinks is the next instruction. Actually, it's the first byte of the string. STRCOMP, having been called, can thus have a pointer to the string by POPping this address off the stack.

The rest of the party is fairly straight up. It inhales each byte of the string, incrementing the pointer as it goes. When it gets a null it knows that it has reached the end of the string. At this cusp of time and space the pointer into the string is actually pointing at the next executable instruction after the null... or, at least, it will be if it's incremented once more. As such, it can be PUSHed back up on the stack and the routine can be returned from in the normal way.

The baud rate program, then, compares the string at 0082H, the command line argument, with a series of imbedded literals starting with the label OVER. It also keeps track of a table pointer in CX, bumping it up as the party proceeds.

The baud rate for the serial port is set by loading two registers in the 8250 serial chip with divisor values. The values are held in a table starting with BD50. The CX register is used to point into the table. When the tottering stack of string compares finally topples over and exudes a result the pointer in CX will be looking at the corresponding baud rate divisor word.

You will note that the help message for this program is printed in the traditional way, using a single INT 21H call. We'll look at an easier trip in the next program.

Dem PC Utility Blues

Radio Waves

In the March 1984 edition of Computing Now! we presented a pair of articles which outlined, respectively, the hardware and software required to receive shortwave radio teletype BAUDOT code and display it in ASCII on an Apple. This was a good trip, and a lot of heads got into it. It was a bit tricky to use on an IBM, however, as, being in 6502, the program always crashes on a blue box.

The RTTY program in this feature is a radio teletype decoder which runs on a PC. It works admirably with the demodulator hardware presented last March, or, if you aren't up for that design, you can wait for a more sophisticated demodulator we'll be running shortly.

There are a lot of similarities between this program and the BAUD program we've just looked at. For one thing, they both set the baud rate of the serial port, although this one has a single set of divisors set for 45.45 baud, the transmission rate for most BAUDOT information. However, this program has two interesting elements, to wit, an in line printing routine and a translation table.

The former is probably the most universally profound. It works in much the same way as does the STRCOMP did a minute ago. The form of its use is

```
CALL   ILPRT
DB     'Oh to be a purple dragon...',0
```

with, once again, the string terminated by a null. The ILPRT routine works in much the same way as does the STRCOMP one, POPping the pointer to the string from the stack. There is, in fact, a little more stack action here because it's necessary to save the pointer onto the stack each time the printing interrupt is called so as to be sure it will be preserved.

The primary purpose of this code is to take BAUDOT data heaved into the serial port and display it as ASCII on the screen. This is a bit tricky, because BAUDOT is a mechanical code. Rather than having been designed to make sense in a numerical space, it was set up to ease the lives of weary teletype designers. BAUDOT makes for a reasonably sensible arrangement of gears and pawls... but a totally unfathomable character set.

The BAUDOT code has five bits, which corresponds to thirty-two unique characters. This doesn't allow for all the letters and ten numbers, let alone any punctuation, so BAUDOT uses a really funky arrangement of two definitions for each character. For example, the character 03 can be either an "A" or a dash.

Which of these it actually turns out to be will be a function of which of the case characters has most recently been sent. If the receiving terminal has had 1FH last, all the characters will show up as letters. If it's had 1BH they'll be figures... numbers and punctuation.

Aside from this weirdness, this program operates with a fairly standard arrangement called a translation table. In many cases... the problem of converting BAUDOT to ASCII being a fine example... there is no obvious numerical relationship between the data one has and the data one wishes to make it into. In this case, one must create a translation or "look up" table.

There are two ways to do this. The one I've used here is the sloppier of the two, but it's a lot easier to see how it works. The program scans through the table two bytes at a time. Each of the odd numbered bytes represents a legal BAUDOT character. If the character which has been received matches an entry in the table the program will back up one byte and take the corresponding ASCII value to replace the BAUDOT code.

There is an easier way in some cases... although it wouldn't have worked well here. We could create a table of the values we want such that the data we have forms a pointer into the table. For example, the BAUDOT code for "A" is 03. We would thus create a table such that 41H, the ASCII code for "A", was the third element in the table. Adding the BAUDOT code to the start of the table would point to the ASCII value.

The BAUDOT series is so fragmented, however, that the table would have been quite large with a lot of gaps.

Snatch Them Routines

These little programs were written for specific applications... you will, undoubtedly, have applications of your own. These things will provide you with a useful framework to hang your own code on.

Of course, there are a lot of things you can do with a BAUDOT terminal... especially if you like really slow telecommunications. Three hundred baud isn't really all that bad after a half an hour at forty-five and a half. Your screen will think it's gone for a permanent nap. All the interrupts will seize up...

Yes, there *is* a source of characters which is slower than an IBM PC keyboard. It damages the mind just thinking about it.

CN!

ing speed, I saw visions of greatness. I could probably turn out modern classics in a few spare hours.

As usual, everything is easier on television than in real life.

My first inkling that everything wouldn't be so smooth came when my eight year old son told me he couldn't get *Chirpee* running. Maybe it's not fair to criticize a technical system because it can't be run by an eight year old but I've found that's a good first test.

As it turned out, he had managed to connect the thing up all right. You merely plug the small printed circuit with an edge connector into the user parallel plug at the back the computer. The manual warns you to plug it in before you turn on your computer. This should be highlighted since shorting any of the connector slots can ruin both the computer and *Chirpee*.

My son managed all this but that's as far as he could go. For the next week, I didn't get much further either. It took a while before all the details worked themselves out.

Utterances

Once you've got the pieces connected, you'll want to do something with the package. ENG includes three sample programs so you won't have to figure out how to write your own. Two are games and one is a "practical" application, a voice created card file. There is also SPEECH GRAPHICS, which I described earlier.

There's a fifth program called Speech Operating System that you'll have to use before anything else is usable. There was something prophetic in that name, SOS.

To do anything with *Chirpee* you have to create speech files called lexicons. Each file contains a command label and a record of you speaking that command. For instance, the command may be "open"... there'll be a record of how you said that word. You can store up to a hundred and twenty eight words in each eight K lexicon. You may create as many lexicons as you have floppies for but only one can be accessed at a time. You would have to make a separate lexicon for each person using a program because differences in voice, accent and pronunciation would be interpreted as completely different words.

When a program runs, *Chirpee* must also be in memory and your program must know to call it. When you speak into the microphone, *Chirpee* tries to match your input with the data on file. If a match occurs, the command label and address are put into memory. Your program must also contain a line telling it to PEEK this location. From here on, the data is treated like any other input to your program. This is all well documented in the user's manual.

You can manipulate the lexicons with SOS. This program gives you a choice of creating, rehearsing or displaying a lexicon. It also will give you a disk directory and copy a lexicon to or from another disk. I'm happy to report that displaying a directory doesn't destroy what's already in memory as happens when you read a disk with Commodore's disk operating system.

The CREATE option is *Chirpee*'s Achilles' heel. Nothing works until you've used it to build a lexicon. This program asks you to enter a number for each command and then a label. Next it asks you to speak that command into the speaker. Then it stores this data and compares it to the last two inputs. If it matches, it asks you if you want to save this information to disk. Then you go on to the next word. This would be great, if it worked.

On the other hand, if there is any difference between how you said the word when you created a lexicon and how you say it in use, *Chirpee* won't recognize it. So too, if there's any background noise, *Chirpee* will interpret that as part of its input. I had to shut off the furnace and humidifier in the room next door before I could get any commands to match. If my wife turned on the water upstairs, *Chirpee* couldn't cope. Creating a lexicon can be a most frustrating experience.

In fairness, ENG admits this is a problem. Their manual contains a whole page on trying to get a good match. All their suggestions are good but still fail too frequently. If you have too much trouble creating a lexicon, you can use the SPEECH GRAPHICS program to analyze your speech. You'll be surprised how your speech profile can change with only minor changes in your voice.

Once you have a lexicon, SOS will let you check it. The REHEARSE routine lets you review the stored speech and your pronunciation. As you speak, the program will display the label that most closely matches your word. If there isn't a high match rate, it's back to the joys of file creation.

DISPLAY lists all the labels in a file along with their command numbers. It makes no effort to display the voice pattern. Make sure you check each lexicon well before using it. You can waste days debugging a perfect program when the fault is with your lexicon.

Once you've got your lexicon, you're in business. The manual is quite explicit in telling you how to interface your programs to the system. After all the trouble I'd had until this point, I opted to use the sample programs. I found more problems.

CARD FILE gives you a taste of how voice input can improve word processing. Battlestar Galactica aside, there's not much likelihood of having your computer print out your speech. Not for now anyway. On the other hand, many word processor commands lend themselves to input from something other than the keyboard, such as Macintosh's mouse, for instance.

I'd certainly like to yell delete at my SOL, the machine I use for word processing and other serious stuff, instead of pressing the control keys. Life would be a lot simpler if I could growl "escape" instead of relying on the key of the same name that drifts in and out of a coma at inconvenient moments. CARD FILE does just this. There are eleven spoken commands including print, scan and scratch.

WORD MIX is a program written in BASIC to show you how this can be done. I don't know how it works because I couldn't get it to run. Perhaps my pronunciation of "aardvark" is too difficult for any machine to catch. However, listing the program should help potential users learn to interface *Chirpee* to a BASIC program.

I did get AERONAUT working well. It's

Chirpee for the 64

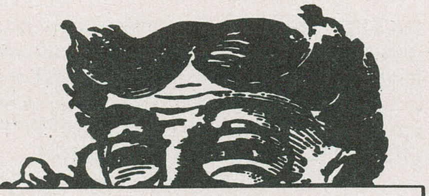
a balloon flight simulation. By yelling, "blast", "puff", "vent", or "panic" you can control the ascent and descent of a hot air balloon. The object is to fly over the trees and power lines and land at several sites.

As I mentioned earlier, I had to change some commands to accomodate my speech pattern. After days of fighting with the SOS program, it was a pleasure to see *Chirpee* work.

The Last Word

In regarding the *Chirpee* package as a whole, I should say that it has potential. Besides the games and rudimentary word processor that came with *Chirpee*, I can think of many more uses. Somebody could take the graphics demonstration program and convert it to speech analysis for speech therapy. Visual feedback would certainly help a person with hearing problems correct speech defects.

Chirpee also offers an untapped wealth of applications possible for the disabled. An inexpensive speech recognition interface coupled to an inexpensive computer could enable a quadriplegic to better cope with life. Think about being able to change your oscilloscope setting without taking your hands off your probes. The possibilities are endless.



Specs

Device:	**Chirpee**
System:	**Commodore 64**
Application:	**Speech Recognition**
Manufacturer:	**ENG Manufacturing, Incorporated, 4304 Saturn Way, Chandler, AZ 85224**
Distributor:	**Romaro Enterprises International Limited, P.O. Box 227, Streetsville Postal Station, Mississauga, Ontario L5M 2B8**
Price:	**$250**

On the other hand, *Chirpee* has definite limitations. Something has to be done to improve the matching of speech patterns. The most obvious way to handle this would be to decrease the sensitivity of the microphone. Unfortunately, the whole solution isn't that simple. I hope ENG can solve this.

The second problem is price. Two hundred and fifty dollars for a practical, working voice activated interface with a library of hardware is cheap. Unfortunately, *Chirpee* doesn't isn't that, at least not yet. Two and a half bills is probably more than most hobbyists are going to pay for something to experiment with.

Until there's this base of experimenters, there won't be a host of application programs.

The third problem is the lack of technical information. The manual doesn't contain schematics or enough information to modify *Chirpee*. I understand the manufacturer's worries about patent protection but the early makers of home computers profited by letting users modify and correct their engineering problems. **CN!**

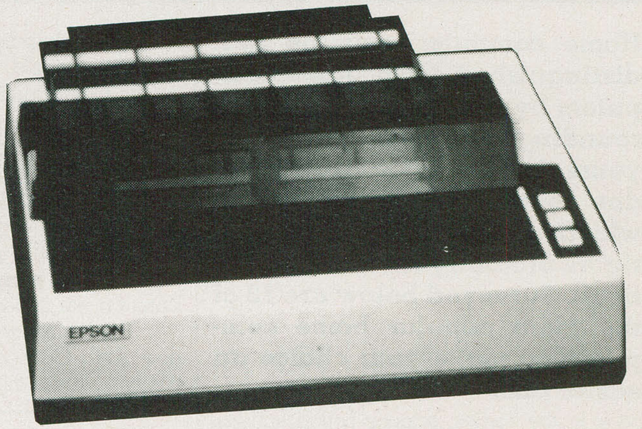
COMPUTER PRESS

The transmission occurred during the weekly radio show *Download*. The Download show transmits free, sponsor supported software to microcomputer owners.

Using a device called the **Shuttle Communicator**, AM or FM stations are able to upload text, computer programs and even pictures to their radio audience. The inexpensive Communicator allows the listener to connect their AM or FM radio to their personal computer.

Data transmission by radio is much faster than using a conventional modem. For example, the picture of the president took 45 *seconds* for the receiving computer to download using the Communicator. The same information would require 24 minutes to download using a conventional telephone-based modem.

Additional information regarding radio downloading can be had from the vice-president of the broadcasting and cable division of the *Microperipheral Corporation*, Mr. Robert E. Lee Hardwick, at 2565 152nd Avenue North East, Redmond, Washington 98052 U.S.A., or by calling (206) 881-7544.



Epson Canada has enhanced the FX series of dot matrix printers with the introductiion of the **FX-80+** and **FX-100+** printers. The FX+ dot matrix printers offer 160 characters per second, with improved 'intelligent' bidirectional logic seeking elements increases print speed 20 per cent over the standard FX models. The FX+ models also offer a button panel that offers immediate access to nine different typestyles...

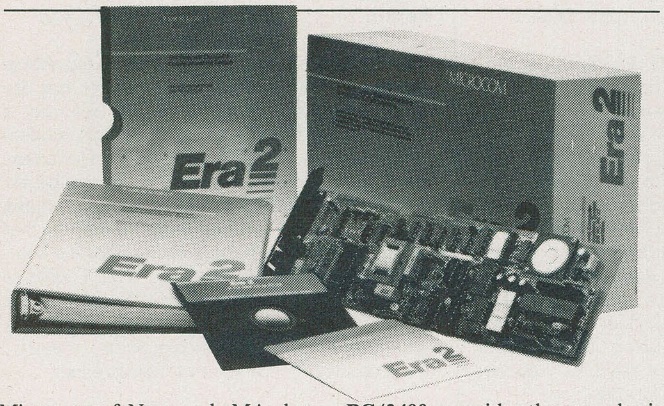
Circle No. 52 on Reader Service Card.

TRW Changeover

WILLOWDALE, ONTARIO – Announced recently by A. Wm. Wilbur, its Vice President and General Manager, *TRW Data Systems* is withdrawing from the marketing and distribution of microcomputers and related peripherals by the end of the first quarter of 1985. This change in the company's activities is meant to allow TRW "...to concentrate its resources in growing with the fast developing maintenance market."

Products presently distributed in Canada by TRW include North Star microcomputers, Wyse terminals and related peripherals. The manufacturers represented by TRW will shortly announce their new Canadian distribution plans. TRW will provide products and support during the transition period to ensure uninterrupted service to dealers and end-users. The company will continue to offer long-term maintenance support on all products.

TRW Data Systems has 28 service locations nation-wide, and is a national leader in the Canadian third-party maintenance business.



Microcom of Norwood, MA, has announced the release of a new line of Hayes compatible 2400 bps, error correcting modems. The **Era 2 PC/2400** system for the IBM PC comes complete with a Hayes compatible 2400 bps internal card modem and software. The system features VT-100, VT-52 and IBM 3101 terminal emulation and single keystroke commands for log on and retrieval of data. The package retails for $799.00 (US). The **PC/2400**, provides the same basic operation as the **Era 2** unit, but in an external design. The retail price is $899.00 (US). The **MacModem 2400** is designed for the Apple Macintosh, and makes use of the mouse for log-on and transfer procedures. The Macintosh sytem features the same terminal emulation as the other products in the new series. It retails of $899.00 (US).

Circle No. 60 on Reader Service Card.

The **Turtle Tot** robot, measuring about nine inches in diameter and weighing in at about five pounds, is a product of *Harvard Associates, Inc.* It can interface to any RS-232 equipped computer and has a 21-foot connecting cable. The robot also draws when used with LOGO, performing at the same time as the screen turtle...

Circle No. 59 on Reader Service Card.

The Bernoulli box by Iomega is being distributed in Canada by *Future Electronics Incorporated.* Plugging into the Apple Macintosh's RS-422 port, the unit uses a single Beta-5 drive for five megabytes of storage. A second five megabyte slave unit is also being produced...

Circle No. 58 on Reader Service Card.

SMT Inc.'s **Real-World Controllers** for the IBM PC and Apple][computers can connect with up to 128 slave cards up to a mile away to sense temperature, pressure or other variables. Communication with remote-slave cards is via pulse-width modulation technique to minimize transmission errors...

Circle No. 57 on Reader Service Card.

To avoid consumer confusion with Georgia-based Hayes Microcomputer Products, Inc., *Hayes Products* have changed their company name to **CH Products**. The popular Mach II and Mach III joysticks for the Apple][computers are manufactured by the renamed company...

Circle No. 56 on Reader Service Card.

Featuring four-point shock mounts, a dynamic spindle brake and an actuator that automatically retracts the read/write heads to a dedicated landing/shipping zone when powered off, the *Shugart* **26-megabyte half-height Winchester** drive offers 20 megabytes formatted storage and uses a closed loop servo system...

Circle No. 55 on Reader Service Card.

An entirely portable and compiled language, **cENGLISH Version 2.30** has been released by its manufacturer, *cLINE Incorporated.* cENGLISH is a comprehensive, fourth generation language with a programming syntax similar to that of dBASE II. It supports multidimensional arrays, procedure function calls, and allows escape into direct C programming...

Circle No. 54 on Reader Service Card.

For the IBM PC programmer that has everything, *Catspaw, Incorporated* is offering **SNOBOL4+**, which encompasses all of mainframe SNOBOL4 except Fortran output formats. The language's extensions include additional string and real functions, SPITBOL compatible operators, binary and random access I/O, built-in sorting, include files, case folding and break key control. Over 100 sample programs and functions are also provided...

Circle No. 53 on Reader Service Card.

Continued on p 78

Survey of Home Computers

Home computers can be best distinguished from other computers by their comfortable surroundings... unless you keep yours in an igloo. Deciding, due to the variety available, which to introduce into the rigors of family life may be difficult. Surveyed below are 13 of the more popular home computers to make your choice an easier one.

Back in the 'forties, science fiction authors and readers alike dreamt of home computers, but they must have had a hard time visualizing them. Computers from that time period... the MARK I, ENIAC and EDSAC... were gigantic behemoths weighing from five to 30 tons and were filled with miles of wiring and scores of vacuum tubes.

Few science fiction enthusiasts could envision a 30 ton ENIAC in their living rooms.

From the 1948 invention of the transistor to the very large scale integration (VLSI) chips of today, computers have come a long way in size, availability and affordability. Home computer kits were on the market around 1974, and the Altair, the first commercially successful microcomputer, was introduced that year. The Ohio Scientific micros began production in 1977, and by 1979 Commodore, Tandy, Apple, Atari and Texas Instruments were producing computers that didn't require their owners to invest in soldering irons and wire wrap. In 1980, Clive Sinclair introduced the ZX-80, a 1K computer barely larger than the span of an adult's hand. Today, you can walk into a department store with less than a hundred dollars in your pocket and leave with a microcomputer with more programming power than the electromechanical giants of the 1940s.

With home computers financially within the reach of practically everyone who wants one, manufacturers are interested in what the potential user wants a home computer for, to better provide for that market. Apple Computer recently released AppleWorks, an integrated software package providing a word processor, database and spreadsheet for its //e and //c computers. Commodore's new *plus/4* computer has three similar programs in ROM, allowing its users to utilize these applications immediately, without first purchasing them, then waiting for them to load into the computer every time they're required.

Microcomputers are different things to different people. The aspiring freelance writer may acquire one to experience the ease of word processing. Parents of pre-schoolers may purchase one to prepare their children for school, then later to supplement their education. Businessmen often get a home computer to communicate with larger ones at work while they're at home. Perhaps best of all, most home computers make fantastic game machines.

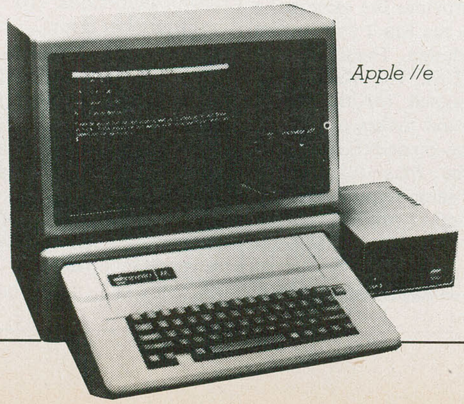
The future looks bright for micros. The new Atari Corporation is rumoured to be readying three different models this spring; an eight-bit, a 16-bit and a 32-bit micro. Commodore's *Amiga* purchase may soon see the fabled 32-bit Amiga on dealer shelves. Apple is working with a recently developed 6502 compatible 16-bit chip that may have a new Apple // computer addressing megabytes of memory.

Despite all the technological advances awaiting around the corner, there's a lot of high technology available right now. In the next few pages we'll be surveying some surprisingly powerful home computer systems. Hold onto your hat... and your joystick.

Apple //e

Operating System: Applesoft BASIC; DOS optional
Processor(s): 6502
RAM: 64K; optional 128K
Printer I/O: Parallel and serial cards available
Disk Drives: Optional; one or two 5 1/4" floppy
Cartridge Port? No
Screen Format: 40x24; optional 80x24
Lowercase? Yes
Graphics: 40x24, 280x192 or 560x192 pixels
Sound: Yes
Colour: Yes
Software Included: BASIC; ProDOS with drive purchase
Manufacturer: Apple Computer Incorporated
Distributor: Authorised Apple Dealers
Suggested Retail: $1495.00; $2395.00 with monitor, drive and extended 80-column card

Description: The Apple //e was introduced as a logical upgrade from the Apple][+. As it stands, the software and hardware base for the //e is immense, but with the addition of a Z-80 co-processor card, the Apple //e can also utilise the large base of CP/M programs and utilities. Unlike the Apple][+, the Apple //e has lowercase characters, can utilise 128K and has, through a design quirk, 'double hires' graphic capabilities of 560 by 192 pixels. The //e has an easily accessible monitor for machine language code entry in hexadecimal format.



Apple //e

Apple //c

Operating System: Applesoft, ProDOS, DOS 3.3
Processor(s): 65C02
RAM: 128K
Printer I/O: Serial
Disk Drives: One 143K floppy; optional 2nd drive
Cartridge Port? No
Screen Format: 40 or 80x24
Lowercase? Yes
Graphics: 40x24, 280x192, 560x192 pixels
Sound: Yes; volume control and headphone jack
Colour: Yes
Software Included: Six tutorial disks
Manufacturer: Apple Computer Incorporated
Distributor: Authorised Apple Dealers
Suggested Retail: $1795.00; $1995.00 with monitor and stand.

Description: A portable version of the Apple //e, the //c has many standard features that are optional with the //e, including 80 columns, a built-in drive, 128K, a serial printer port, modem port, joystick port, RGB port and a port for an external drive. Unlike the //e, the //c cannot be expanded from within, so unless a manufacturer develops a serial Z-80 add-on, it's unlikely that CP/M software can be implemented for the computer. Due to the nature of the CMOS 65C02 processor, some Apple software doesn't operate properly or at all with the //c. Many software houses whose software is incompatible, however, are producing //c versions of their product.

Apple //c

Atari 600XL

Operating System: BASIC
Processor(s): 6502C
RAM: 16K
Printer I/O: Serial
Disk Drives: Optional disk drive or cassette recorder
Cartridge Port? Yes
Screen Format: 40x24
Lowercase? Yes
Graphics: 320x192 pixels; 11 graphics modes
Sound: Yes
Colour: Yes
Software Included: BASIC
Manufacturer: Atari
Distributor: Many computer and department stores
Suggested Retail: $99.99

Description: The Atari 600XL is an upgrade from the 400 computer which had a flat membrane keyboard. The software base for the Atari line of computers is large, with an impressive array of games. The only compatibility restriction with the Atari line at present is memory. A 17K program won't run on a 600XL, but will operate properly on an 800XL. Memory expansion is available for the 600XL. Some features of the 600XL include a choice of 256 colours (128 of which may be displayed on the screen simultaneously), a help key, five text modes and four independent sound channels with a range of three and one-half octaves.

Atari 800XL

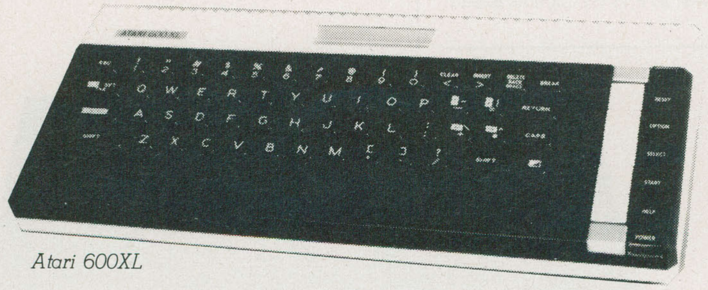
Operating System: BASIC
Processor(s): 6502C
RAM: 64K
Printer I/O: Serial
Disk Drives: Optional disk drive or cassette recorder
Cartridge Port? Yes
Screen Format: 40x24
Lowercase? Yes
Graphics: 320x192 pixels; 11 graphics modes
Sound: Yes
Colour: Yes
Software Included: BASIC
Manufacturer: Atari
Distributor: Many computer and department stores
Suggested Retail: $199.99

Description: Released in late 1983 as a replacement to the Atari 800, the 800XL looks the same and performs the same as the 600XL with the exception of having three times as much memory. The 800XL has an international character set and five text modes, as does the 600XL.

CoCo2

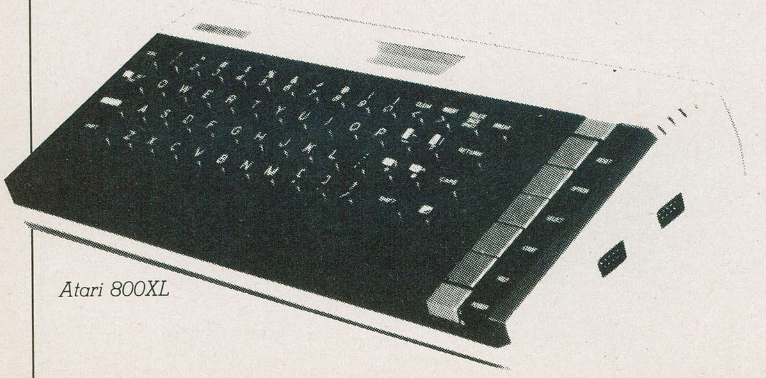
Operating System: BASIC, Extended BASIC and/or Color DOS
Processor(s): 6809E
RAM: 16, 32 or 64K
Printer I/O: Serial
Disk Drives: Optional 5 1/4" 156K floppy
Cartridge Port? Yes
Screen Format: 32x16
Lowercase? No
Graphics: 256x192 pixels
Sound: Yes
Colour: Yes
Software Included: BASIC
Manufacturer: Tandy Electronics
Distributor: Local Radio Shacks
Suggested Retail: $189.00 16K Standard; $249.00 16K Extended; $349.00 64K Extended

Description: CoCo2, short for Color Computer 2, was recently released as a replacement for the original CoCo which had flat, rectangular keys. The CoCo2 features a full typewriter–like keyboard, an interrupt timer and its graphics capabilities range from 32 by 64 pixels in eight colours to 256 by 192 pixels in two colours with three intermediate formats. The 16K Extended BASIC CoCo2 has an additional 8K ROM that allows PEEK, POKE and USR commands, full–featured editing and tracing, and extended graphics handling from BASIC. The 64K Extended BASIC model is similar to the 16K Extended model, though only 32K of its 64K is accessible from BASIC unless Color DOS is implemented. The CoCo2's software base, while not as expansive as that of Apple or Atari, is nonetheless impressive and more than adequate for home use.



Atari 600XL

CoCo2



Atari 800XL

Survey of Home Computers

Coleco Adam

Operating System: SmartBASIC

Processor(s): Z-80A, others

RAM: 80K

Printer I/O: Includes letter-quality printer

Disk Drives: Two stringy tape drives

Cartridge Port? Yes

Screen Format: 36x25

Lowercase? Yes

Graphics: 256x192 pixels

Sound: Yes

Colour: Yes

Software Included: SmartBASIC, word processor, game

Manufacturer: Coleco Industries

Distributor: Coleco

Suggested Retail: $499.00

Description: The Coleco Adam system consists of a computer, a printer and two joysticks. A similar system is available for Colecovision game machine owners than upgrades their machine into an Adam system. The letter quality printer that accompanies the Coleco Adam system makes the computer of interest to prospective purchasers: most letter quality printers cost more than the entire computer system. As the Adam's power supply is integrated into the printer, however, it's necessary to have the printer beside the computer during operation. SmartBASIC, a cassette-loaded operating system, emulates Applesoft, though CALL statements seem to be ignored. Adam's word processor is in ROM. Though recently discontinued by Coleco in the 'States, the Adam is hardware compatible to the Colecovison, and can use all Colecovision game cartridges. Colecovision support is still being continued by the company, and software support for the Adam is still continuing by Coleco Canada.

Commodore 16

Operating System: BASIC

Processor(s): 8501 (6502 compatible)

RAM: 16K

Printer I/O: Serial

Disk Drives: Optional 5 1/4" floppy

Cartridge Port? Yes

Screen Format: 40x25

Lowercase? Yes

Graphics: 320x200 pixels

Sound: Yes

Colour: Yes

Software Included: BASIC

Manufacturer: Commodore Business Machines

Distributor: Authorised Commodore Dealers

Suggested Retail: $199.95

Description: Commodore's new low-end computer has 16K of RAM (12K accessible from BASIC), an upgraded BASIC with extended graphics and disk commands, admirable colour control, a built-in assembler/monitor for machine language programming, a reset button, programmable function keys and a help key. The 16's cassette and joystick ports differ from those of the Vic and 64, so these peripherals are not interchangeable. Similarly, Commodore 64 or Vic-20 cartridges won't interface with the 16. The 16 has no user port, so Vic and 64 interfaces, such as modems and real-world controllers aren't compatible with the 16. Excepting the lack of a user port, the 16 appears to be a superior machine to the Vic-20, and third-party software and hardware manufacturers can be expected to take a strong interest in it in the months ahead.

Commodore 64

Operating System: BASIC

Processor(s): 6510 (6502 compatible)

RAM: 64K

Printer I/O: Serial

Disk Drives: Optional 5 1/4" floppy

Cartridge Port? Yes

Screen Format: 40x25

Lowercase? Yes

Graphics: 320x200 pixels; sprites

Sound: Yes

Colour: Yes

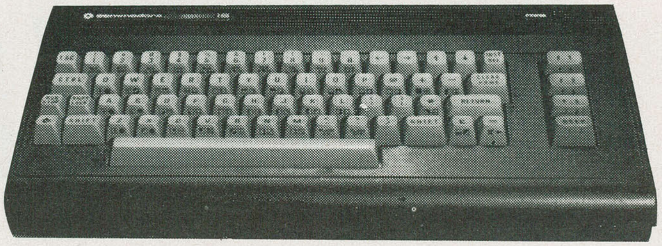
Software Included: BASIC

Manufacturer: Commodore Business Machines

Distributor: Authorised Commodore Dealers

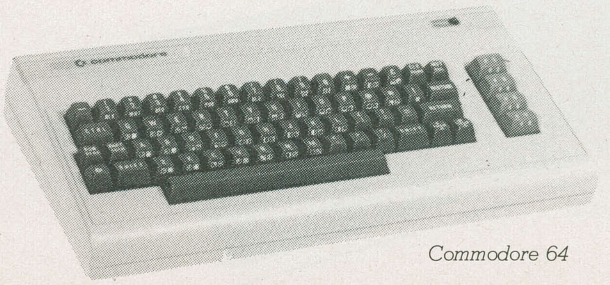
Suggested Retail: $429.95

Description: Introduced in 1982 to complement the Vic-20, the Commodore 64 has become that company's best seller with a very large line of support from both Commodore and other software and hardware manufacturers. Features include SID, a 6581 synthesizer chip allowing programmatic control over ADSR, waveform and volume, 64K of memory (38K accessible from BASIC), eight sprites, redefinable characters and the ability to have 16 colours on-screen simultaneously. 80-column interfaces are available for word processing, as are Z-80 cartridges for CP/M usage.



Commodore 16

Coleco Adam

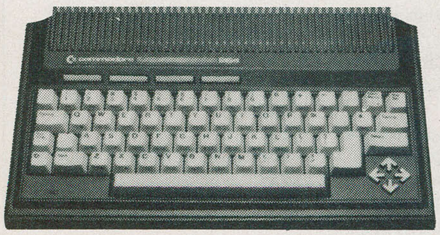


Commodore 64

Commodore plus/4

Operating System:	BASIC
Processor(s):	8501 (6502 compatible)
RAM:	64K
Printer I/O:	Serial
Disk Drives:	Optional 5 1/4" floppy
Cartridge Port?	Yes
Screen Format:	40x25
Lowercase?	Yes
Graphics:	320x200 pixels
Sound:	Yes
Colour:	Yes
Software Included:	BASIC, word processor, spreadsheet, filer
Manufacturer:	Commodore Business Machines
Distributor:	Authorised Commodore Dealers
Suggested Retail:	$529.95

Description: The plus/4, introduced alongside the Commodore 16 in late 1984 has everything the 16 features and more. About 60K of its 64K is accessible from BASIC, an RS-232 communications interface is provided, as are separate cursor keys. The three programs in ROM are the most used applications in home computing, so plus/4 users won't have to purchase spreadsheet, word processing and filing software. Spreadsheet figures can be displayed in text graphs. Unlike the Commodore 64, the plus/4 has neither a SID chip, nor sprite capabilities. Like the 16, however, the plus/4 has two-voice square wave sound and up to 16 colours with eight levels of luminance. Both the plus/4 and the 16 are reviewed elsewhere in this issue.



Commodore plus/4

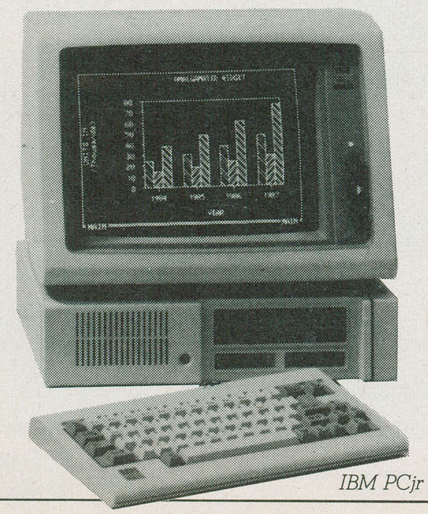
Circle No. 7 on Reader Service Card

Survey of Home Computers

IBM PCjr

Operating System: BASIC; MS-DOS 2.x enhanced
Processor(s): 8088
RAM: 64K; 128K enhanced
Printer I/O: Serial
Disk Drives: Optional 5 1/4" floppy; 1 DSDD 5 1/4" floppy enhanced model
Cartridge Port? Yes
Screen Format: 40x24 or 80x24
Lowercase? Yes
Graphics: 320x200 or 640x200 pixels
Sound: Yes
Colour: Yes
Software Included: Software in ROM
Manufacturer: International Business Machines
Distributor: Authorised IBM dealers
Suggested Retail: $998.00 Entry model; $1569.00 Enhanced model

Description: Announced in November 1983, the IBM PCjr joined IBM's personal computer line in early 1984. Though a powerful computer in its own right, critics had a field day over the cordless keyboard, which then had raised 'chiclet' rectangular keys. IBM corrected the situation with an impressive typewriter-style cordless keyboard which is now standard on all units. The entry level model PCjr is a cassette- and cartridge-based system with numerous programs available on cartridge, including Lotus 1-2-3. A program called 'Keyboard Adventure' is built into the PCjr's ROM. The entry level model is easily upgraded into the enhanced model which features 128K RAM and a double-sided, double-density disk drive. Further upgrading of both models is possible: the addition of RAM packs up to 512K, a second drive or a hard disk. As the December 24, 1984 issue of *Time* noted, the PC-jr has about 40 per cent compatibility with the software base established for the IBM PC, but similar to the case of the Apple //c, many PC software manufacturers are producing PCjr compatible versions of their products.



IBM PCjr

Radio Shack Model 100

Operating System: BASIC
Processor(s): CMOS 80C85
RAM: 8K or 24K; expansion up to 32K
Printer I/O: Parallel and serial
Disk Drives: Optional expansion and 5 1/4" floppy
Cartridge Port? No; ROM socket
Screen Format: 40x8 LCD
Lowercase? Yes
Graphics: 240x64 pixels
Sound: Yes
Colour: No
Software Included: BASIC, word processer, filer, appointments, telecommunications.
Manufacturer: Tandy Electronic
Distributor: Local Radio Shacks
Suggested Retail: $799.00 8K; $1099.00 24K

Description: Weighing less than four pounds, the Model 100 has a full-size typewriter-style keyboard with eight programmable function keys, a bar code reader port, rechargeable batteries (an AC adaptor is optional), and a built-in modem. The modem, operated through the TELCOM program in ROM, is direct-connect and features auto-dialling. The Model 100 is about the size of an 8 1/2" by 11" piece of paper, and is two inches thick. While the software base for the Model 100 is not as large as that of Apple, Atari, IBM or the Commodore 64, the Model 100 has a number of business applications written for it which may make it ideal for the writer or businessman at home or, owing to its portability, in transit.

Radio Shack Model 100

Addresses: Apple Canada Incorporated, 875 Don Mills Road, Don Mills, Ontario M3C 1V9 (416) 444-2531 * Radio Shack/Tandy Electronics Limited, 279 Bayview Drive, Barrie, Ontario L4M 4W5 (416) 728-6242 * Coleco Industries (Canada), 5149 Bradco, Mississauga, Ontario L4W 2A6 (416) 624-3401 * Commodore Business Machines LImited, 3370 Pharmacy Avenue, Agincourt, Ontario M1W 2K4 (416) 499-4292 * IBM Canada Limited, 3500 Steeles Avenue East, Markham, Ontario L3R 2Z1 (416) 474-2053 * Spectravideo Canada, 2913 Lakeshore Boulevard West, Toronto, Ontario M8V 1J3 (416) 252-4550

Spectravideo SV-318

Operating System: BASIC
Processor(s): Z-80A
RAM: 32K
Printer I/O: Expansion bus
Disk Drives: Cassette driven; optional 5 1/4" floppy
Cartridge Port? Yes
Screen Format: 40x24; optional 80-column card
Lowercase? Yes
Graphics: 256x192 pixels; 32 sprites
Sound: Yes
Colour: Yes
Software Included: BASIC, three games
Manufacturer: Spectravideo
Distributor: Spectravideo Canada
Suggested Retail: $299.00

Description: Introduced in late 1983, the Spectravideo SV-318 was a bit of an oddity... instead of cursor keys, it had a built-in joystick. More recently, the computer's design has been the catalyst for Microsoft's MSX standard, which, employed in Japan and Europe, has recently been introduced to North America. Spectravideo should have a new MSX computer available by the time you read this. The SV-318 has flat rubber keys, ten programmable function keys, and, with memory expansion and the addition of a disk drive, is CP/M compatible.

Spectravideo SV-328

Operating System: BASIC, or CP/M with disk system
Processor(s): Z-80A
RAM: 80K
Printer I/O: Optional expander available
Disk Drives: Cassette driven. Optional 5 1/4" floppy
Cartridge Port? Yes
Screen Format: 40x24. Optional 80 column cartridge.
Lowercase? Yes
Graphics: 256x192; 32 sprites
Sound: Yes
Colour: Yes
Software Included: BASIC
Manufacturer: Spectravideo
Distributor: Spectravideo Cañada
Suggested Retail: $499.00

Description: Also released in late 1983, the SV-328 is similar in most respects to the SV-318, though the joystick has been replaced with individual cursor keys and a numeric keypad, the keyboard is 'typewriter' quality, and the unit has 80K of RAM. The SV-328 is 100 per cent hardware and software compatible to the SV-318, though naturally a program more than 32K in length won't operate on an unexpanded SV-318. With the addition of a disk system, the SV-328 operates under the CP/M operating system.

CN!

Colour Computer Print Utility

The Radio Shack Colour Computer can be interfaced to a number of interesting peripherals. Some, like the popular Gemini printers, require a little extra software to make them feel at home.

by David Caswell

Adding a printer to your system is a worthwhile effort. It will offer you access to applications you've probably never thought of before.

Contemporary dot matrix printers, such as the Star Micronics Gemini machines we'll be looking at in this feature, offer a wide variety of features which can be accessed through software control. While in most applications these things can be selected and manipulated manually... by just typing in the right sequences of control characters... it's usually the case that the control strings are anything but mnemonic, and one quickly finds one's self buried the user's manual every five minutes.

There is, of course, an easier way. Consider the print formatter, the program accompanying this article.

The Code

This program is designed for use with a Colour Computer having from sixteen to sixty-four K of memory. It is presently set up to operate with a Gemini 10X printer, but can easily be altered to suit the requirements of other hardware.

The program itself is written in position independent 6809 machine language which is loaded into the last three hundred and seventeen available memory locations in RAM and protected from BASIC. This allows BASIC programs to be loaded and changed while the print formatter is in place.

It will operate with either Colour BASIC or Extended Colour BASIC.

To load the print formatter, enter and RUN the basic driver program shown in the first listing. Make a note of the start address, the BASIC call address and the finish address displayed by the driver program. You may want to save a copy of the machine language routine by typing

CSAVEM "PRNTFRM", *start address, finish address, start address.*

Once positioned at the top of memory, the print formatter can be used in two ways. It can be either a standalone utility or a BASIC subroutine. When used on its own by typing

Colour Computer Print Utility

The BASIC driver

```
10 REM BASIC DRIVER FOR PRINT FORMATTER
20 M=PEEK(35)
30 IF M>63 THEN CLEAR 100.32450 ELSE CLEAR 100.16066
40 M=PEEK(35)
50 IF M>63 THEN AD=32451 ELSE AD=16067
55 CLS:PRINT"LOADING PROGRAM"
60 FOR L=AD TO AD+316
70 READ DT
80 POKE L.DT
90 NEXT L
100 PRINT:PRINT"LOADED INTO HIGH MEMORY"
102 PRINT:PRINT"START ADDRESS =":AD
103 PRINT:PRINT"BASIC CALL ADDRESS =":AD+44
104 PRINT:PRINT"FINISH ADDRESS =";AD+316
110 PRINT:PRINT"TEST BY TYPING 'EXEC":AD;"'"
120 END
150 DATA 15. 111. 189, 169, 40, 48, 141, 0, 160, 49
160 DATA 141, 0, 134, 198, 11, 238, 161, 223, 136. 141
170 DATA 53, 90, 38, 247, 173, 159, 160, 0, 39, 250
180 DATA 129, 48, 37, 246, 129, 57, 34, 242, 128, 48
190 DATA 31, 137, 32, 3, 189, 179, 237, 48, 141, 0
200 DATA 86, 58, 230, 132, 48, 141, 0, 29, 58, 134
210 DATA 254, 151, 111, 141, 9, 134, 7, 173, 159, 160
220 DATA 2, 15, 111, 57, 166, 128, 129, 255, 39, 6
230 DATA 173, 159, 160, 2, 32, 244, 57
235 REM PRINT CONTROL
240 DATA 27. 64, 255, 27, 66, 2, 255, 27, 66, 3, 255, 27, 52
250 DATA 255, 14, 255, 27, 45, 1, 255, 27, 70, 255
260 DATA 27, 83, 0, 255, 27, 83, 1, 255, 27, 71 , 255
265 REM RESERVED MEMORY
270 DATA 128. 128, 128, 128, 128, 128, 0, 192, 237
280 DATA 233, 237, 233, 237, 233, 233, 237
285 REM TABLE OFFSETS
290 DATA 0, 3, 7, 11, 14, 16, 20, 23, 27, 31
300 DATA 4, 69, 4, 129, 4, 145, 4, 193, 4, 209, 5, 1, 5
310 DATA 17, 5, 65, 5, 81, 5, 129, 5, 145, 83
320 DATA 69, 76, 69, 67, 84, 32, 80, 82, 73, 78
330 DATA 84, 69, 82, 32, 79, 80, 84, 73, 79, 78
340 DATA 255, 48, 41, 32, 82, 69, 83, 69, 84, 255
350 DATA 49, 41, 32, 49, 50, 32, 80, 73, 84, 67
360 DATA 72, 255, 50, 41, 32, 49, 55, 32, 80, 73
370 DATA 84, 67, 72, 255, 51, 41, 32, 73, 84, 65
380 DATA 76, 73, 67, 83, 255, 52, 41, 32, 69, 78
390 DATA 76, 65, 82, 71, 69, 68, 255, 53, 41, 32
400 DATA 85, 78, 68, 69, 82, 76, 73, 78, 69, 255
410 DATA 54, 41, 32, 69, 77, 80, 72, 65, 83, 73
420 DATA 90, 69, 68, 255, 55, 41, 32, 83, 85, 80
430 DATA 69, 82, 83, 67, 82, 73, 80, 84, 255, 56
440 DATA 41, 32, 83, 85, 66, 83, 67, 82, 73, 80
450 DATA 84, 255, 57, 41, 32, 68, 66, 76, 69, 32
460 DATA 83, 84, 82, 73, 75, 69, 255
```

EXEC, start address

...a menu is displayed outlining ten commonly used printer formats. Upon receiving your selection, the program sends the appropriate control sequence to the printer and sounds the printer bell, acknowledging the command. You are again returned to BASIC in the command mode as before.

If used as a BASIC subroutine, the routine is called using the USR statement. You must include a DEF USR = X where X is the BASIC call address as given in the driver program. The argument of the USR(N) statement indicates your selection in the range of zero to nine, corresponding to the desired printer option. Again the printer bell is sounded and control is returned to the next BASIC statement in your program.

You may then continue on in the BASIC program sending output to the printer using the most recently specified format.

Any BASIC program which frequently changes the print style will benefit from this simple subroutine by eliminating the clutter resulting from many print control statements. The only difference between the standalone version and the subroutine version is that in case of the latter there is no menu displayed and the option to be selected is contained in the subroutine call as an argument.

Power of Print

The program can be easily modified to handle another type of printer by changing values in two tables. The second contains the

The assembler source

```
* PRINT FORMATTER
* BY DAVID CASWELL
* MARCH 1984.

         ORG     $7D00    SELECT HI MEM
CHRIN    EQU     $A000    DEFINE INPUT
CHROUT   EQU     $A002    DEFINE OUTPT
CLEAR    EQU     $A928    DEVINE SCRN CLEAR
CURSOR   EQU     $88
DEVNUM   EQU     $6F
INTCNV   EQU     $B3ED
START    CLR     DEVNUM   OUTPUT TO SCREEN
         JSR     CLEAR    CLEAR SCREEN
*
* DISPLAY MENU
*
         LEAX    MSG.PCR
         LEAY    CURPOS.PCR       GET TABLE LOCN
         LDB     #11              PRINT COUNTER
MENU     LDU     .Y++     GET CURSOR POSN
         STU     CURSOR   MOVE CURSOR
         BSR     OUTPUT   PRINT TEXT
         DECB             FINISHED MENU?
         BNE     MENU     CONTINUE
*
* MENU SELECTION
*
KEYIN    JSR     [CHRIN]  CHECK KEYBOARD
         BEQ     KEYIN    LOOP TILL FOUND
         CMPA    #$30     TEST INPUT
         BLO     KEYIN    IF NOT
         CMPA    #$39     NUMERIC
         BHI     KEYIN    REPEAT
         SUBA    #$30     CONVERT CHR TO NUMBER
         TFR     A.B      PUT INPUT IN B
         BRA     SELECT   SKIP BASIC ENTRY
BASIN    JSR     INTCNV BASIC ENTRY
SELECT   LEAX    TABOFF.PCR       GET OFFSET FOR
         ABX              DATA LOOKUP
         LDB     .X       RESULT TO B
         LEAX    TABLE.PCR        FIND START
         ABX              OF PRINT DATA
*
* WRITE TO PRINTER
*
         LDA     #-2
         STA     DEVNUM   OUTPUT TO PRINTER
         BSR     OUTPUT   SEND CNTRL CHARS
         LDA     #7       SOUND
         JSR     [CHROUT] BELL
         CLR     DEVNUM   RESET SCREEN
         RTS              RETURN TO BASIC
OUTPUT   LDA     .X+
         CMPA    #$FF
         BEQ     RETURN
         JSR     [CHROUT]
         BRA     OUTPUT
RETURN   RTS
TABLE    FCB     27       DATA FOR INITIALIZATION
         FCB     64
         FCB     $FF
         FCB     27       12 PITCH
         FCB     66
         FCB     2
         FCB     $FF
         FCB     27       17 PITCH
         FCB     66
         FCB     3
         FCB     $FF
         FCB     27       ITALICS
         FCB     52
         FCB     $FF
         FCB     14       ENLARGED
         FCB     $FF
         FCB     27       UNDERLINE
         FCB     45
         FCB     1
         FCB     $FF
         FCB     27       EMPHASIZED
         FCB     70
         FCB     $FF
         FCB     27       SUPERSCRIPT
```

```
00900          FCB     83
00910          FCB     0
00920          FCB     $FF
00930          FCB     27           SUBSCRIPT
00940          FCB     83
00950          FCB     1
00960          FCB     $FF
00970          FCB     27           DOUBLE-STRIKE
00980          FCB     71
00990          FCB     $FF
01000          RMB     $10          RESERVE FOR CHANGES
01010 TABOFF   FCB     0
01020          FCB     3
01030          FCB     7
01040          FCB     11
01050          FCB     14
01060          FCB     16
01070          FCB     20
01080          FCB     23
01090          FCB     27
01100          FCB     31
01110 CURPOS   FDB     $0445
01120          FDB     $0481
01130          FDB     $0491
01140          FDB     $04C1
01150          FDB     $04D1
01160          FDB     $0501
01170          FDB     $0511
01180          FDB     $0541
01190          FDB     $0551
01200          FDB     $0581
01210          FDB     $0591
01220 MSG      FCC     "SELECT PRINTER OPTION"
01230          FCB     $FF
01240          FCC     "0) RESET"
01250          FCB     $FF
01260          FCC     "1) 12 PITCH"
01270          FCB     $FF
01280          FCC     "2) 17 PITCH"
01290          FCB     $FF
01300          FCC     "3) ITALICS"
01310          FCB     $FF
01320          FCC     "4) ENLARGED"
01330          FCB     $FF
01340          FCC     "5) UNDERLINE"
01350          FCB     $FF
01360          FCC     "6) EMPHASIZED"
01370          FCB     $FF
01380          FCC     "7) SUPERSCRIPT"
01390          FCB     $FF
01400          FCC     "8) SUBSCRIPT"
01410          FCB     $FF
01420          FCC     "9) DBLE STRIKE"
01430 END      FCB     $FF
01440          END     START
```

assembly language source code. The tables to be changed for non–Gemini printers are TABLE and TABOFF.

If you have an assembler, you may wish to enter the assembly listing, changing the table values as you go and then reassembling. Otherwise, changes can be made in the DATA statements of the driver program.

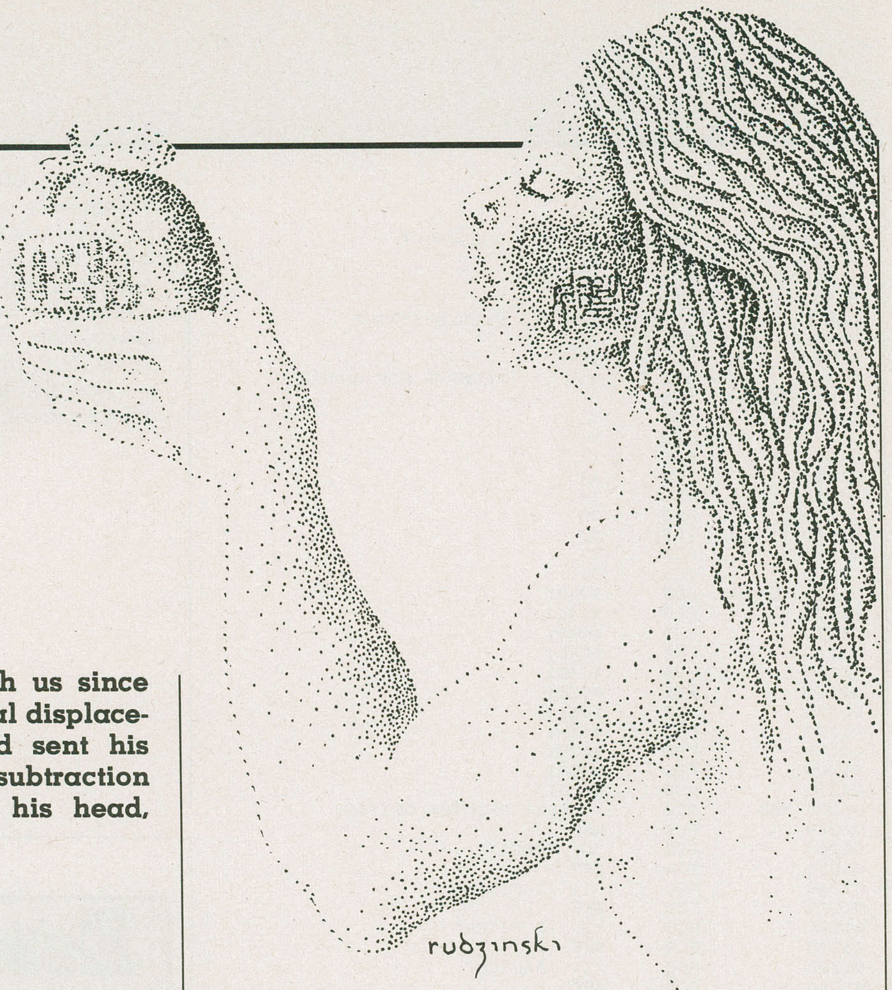
The data for the format control from TABLE is contained between lines 660 and 990 in the assembly language listing and between lines 240 and 280 in the BASIC driver. In each case, a delimiter of $FF is used to separate the various control sequence groups. Following these groups, sixteen extra memory locations are reserved to accommodate possible larger sequence groups for another printer.

In the BASIC driver, this reserved memory is contained in lines 270 and 280. It should be noted that whatever printer is being used, lines 240 to 280 must contain exactly fifty values. In the case of the Gemini, the first thirty–four are real data and the remaining sixteen are space fillers.

If the table of control sequences is changed as described above, the table of offsets must also be changed. The ten values in the offset table indicate the number of memory locations away from the top of the control sequence table that the selected control sequence group is to be found. For example there is no offset for option zero as the sequence group is at the top of the table. If option two were selected, the offset would be figured as the number of characters for option zero plus the number for option one plus the two delimiters... which works out to seven in the Gemini table. This table is found in the assembly listing beginning at line 1010, label TABOFF and in the BASIC driver at line 290.

Now that the program is in memory and modified as required, you can put aside that users manual. **CN!**

Apple Encryption



Information encryption's been with us since Julius Caesar added an alphabetical displacement value to his messages and sent his couriers running. Hexadecimal subtraction would have left him scratching his head, though...

by John Rudzinski

Everybody's got secrets. From governments and multinational corporations to Bertha the bag lady, everyone has something they'd rather not share with either the competition or the general populace as a whole. As alluded to in the introduction, ways and means of protecting information have been around since the days of the Roman Empire.

Julius Caesar befuddled foes who intercepted his dispatches by adding a single offset to all the letters in his messages. An offset of three would be responsible for the word "hail" appearing in a message as "kdlo", an unusual uttering of respect. Similarly, an offset of negative one would produce "HAL" from "IBM", something which mister Clarke maintains he never thought of.

I know... but there are still people who haven't *seen* 2001.

Encryption methods grew in complexity from from this simple procedure. Reversed words were popular for a time. A solid block of reversed words peppered with random spaces can be suitably confusing. Reverse the alphabet, making A represent Z and Z stand for A, and messages have a unique flavour all to their own. They're also difficult to comprehend unless you lay a reversed alphabet underneath a normal one and translate the message from there.

During World War II substitution was common. Books that held the equates were distributed to the top brass and agents. The one weakness to this approach was that if an agent was caught with the book in his possession, decryption of subsequent transmissions or written messages by interested enemy parties was greatly simplified, to say the least.

In theory, anything that can be encrypted can be decrypted by someone determined to do so. The safest methods of encryption are therefore those which leave little for code crackers to work with.

If you write a program that, say, subtracts thirteen from all your data bytes and rotates the bits to the right a number of times, you'll admittedly have a pile of gibberish when you've finished the encoding. However, you'll also have your encryption program laying around. Anyone getting hold of it could determine exactly what you did to encrypt your bytes and write a program to do exactly the same thing in reverse, decrypting them perfectly. If, however, the value that you subtracted isn't resident in the program, the thief would have to guess your key... perhaps two hundred and fifty-five times.

Crypt to My Lou

The two programs accompanying these words are written in 6502 assembler for the Apple][. One's an encryptor and the other, suitably, puts everything back together again. The programs will only work on binary files or text residing in memory. In short, if you can't BLOAD or BSAVE your files in some manner, turn the page.

The majority of Apple word processors, database programs and spreadsheets store data files in binary format, though. Read on, MacDuff.

To get your binary bytes to do the metamorphosis mambo you'll need the inherent bytes of program two and a binary file or program of your own devising.

First, BLOAD *Binary File Encryption*. It's presently assembled at \$8000, but can, with care, be relocated if that address conflicts with your program. The only bytes you'll need to change rest at line number 145 on the listing. Change 74 80 (\$8074) starting at address \$8192 to the low and high bytes that you'd prefer.

BLOAD your program next. It's necessary that you know

Listing 1, the decryptor

```
:ASM
                   *
                   *
                   *      APPLE BINARY FILE DECRYPTION
                   *
                   *         COPYRIGHT (C) 1984
                   *
                   *           JOHN RUDZINSKI
                   *
                   *
                   ADLO     EQU  $05
                   ADHI     EQU  $06
                   COUNTER  EQU  $07
                   BASIC    EQU  $E003
                   BUFF     EQU  $200
                   BUFF2    EQU  $300
                   GET      EQU  $FD0C
                   HOME     EQU  $FC58
                   INPUT    EQU  $FD6A
                   MOVSTR   EQU  $E5E2
                   PRBYTE   EQU  $FDDA
                   STROUT   EQU  $DB3A
                   *
8000: 18                    CLC
8001: 90 5B                 BCC  START
8003: C5 CE D4     MSG1     ASC  "ENTER START ADDRESS: "00
8006: C5 D2 A0 D3 D4 C1 D2 D4
800E: A0 C1 C4 C4 D2 C5 D3 D3
8016: BA A0 00
8019: 8D 8D        MSG2     HEX  8D8D
801B: C1 C4 C4              ASC  "ADDRESS: $"00
801E: D2 C5 D3 D3 BA A0 A4 00
8026: 8D           MSG3     HEX  8D
8027: C3 CF CE              ASC  "CONFIRM (Y/N): "00
802A: C6 C9 D2 CD A0 A8 D9 AF
8032: CE A9 BA A0 00
8037: 8D 8D        MSG4     HEX  8D8D
8039: C5 CE D4              ASC  "ENTER PASSWORD:"00
803C: C5 D2 A0 D0 C1 D3 D3 D7
8044: CF D2 C4 BA 00
8049: 8D 8D        MSG5     HEX  8D8D
804B: D0 C1 D3              ASC  "PASSWORD: "00
804E: D3 D7 CF D2 C4 BA A0 00
8056: C4 CF CE     MSG6     ASC  "DONE."8D8D00
8059: C5 AE 8D 8D 00
805E: 20 58 FC     START    JSR  HOME
8061: A9 03                 LDA  #<MSG1
8063: A0 80                 LDY  #>MSG1
8065: 20 3A DB              JSR  STROUT
8068: A9 A4                 LDA  #$A4
806A: 85 33                 STA  $33
806C: 20 6A FD              JSR  INPUT
806F: A2 00                 LDX  #$00
8071: E0 04        GETBYTE  CPX  #$04
8073: F0 21                 BEQ  ENSURE
8075: BD 00 02              LDA  BUFF,X
8078: C9 B0                 CMP  #$B0
807A: 30 E2                 BMI  START
807C: C9 BA                 CMP  #$BA
807E: 10 07                 BPL  HEXLTR
8080: 9D 00 03              STA  BUFF2,X
8083: E8                    INX
8084: 18                    CLC
8085: 90 EA                 BCC  GETBYTE
8087: C9 C1        HEXLTR   CMP  #$C1
8089: 30 D3                 BMI  START
808B: C9 C7                 CMP  #$C7
808D: 10 CF                 BPL  START
808F: 9D 00 03              STA  BUFF2,X
8092: E8                    INX
8093: 18                    CLC
8094: 90 DB                 BCC  GETBYTE
8096: A9 19        ENSURE   LDA  #<MSG2      ;NOT SURE
8098: A0 80                 LDY  #>MSG2
809A: 20 3A DB              JSR  STROUT
809D: A2 00                 LDX  #$00
809F: E0 04        LOGIC    CPX  #$04
80A1: F0 24                 BEQ  EEYORE
80A3: BD 00 03              LDA  BUFF2,X
80A6: C9 BA                 CMP  #$BA
80A8: 10 18                 BPL  LTR
80AA: E9 AF                 SBC  #$AF
80AC: E0 00        COMPARE  CPX  #$00
80AE: F0 0B                 BEQ  SHIFT
80B0: E0 02                 CPX  #$02
80B2: F0 07                 BEQ  SHIFT
80B4: 9D 00 03     STORE    STA  BUFF2,X
80B7: E8                    INX
80B8: 18                    CLC
80B9: 90 E4                 BCC  LOGIC
80BB: 0A           SHIFT    ASL
80BC: 0A                    ASL
80BD: 0A                    ASL
80BE: 0A                    ASL
80BF: 18                    CLC
80C0: 90 F2                 BCC  STORE
80C2: E9 B7        LTR      SBC  #$B7
80C4: 18                    CLC
80C5: 90 E5                 BCC  COMPARE
80C7: AD 00 03     EEYORE   LDA  BUFF2
80CA: 4D 01 03              EOR  BUFF2+1
80CD: 48                    PHA
80CE: AD 02 03              LDA  BUFF2+2
80D1: 4D 03 03              EOR  BUFF2+3
80D4: 8D 00 03              STA  BUFF2
80D7: 68                    PLA
80D8: 8D 01 03              STA  BUFF2+1
80DB: 85 06                 STA  ADHI
80DD: 20 DA FD              JSR  PRBYTE
80E0: AD 00 03              LDA  BUFF2
80E3: 85 05                 STA  ADLO
80E5: 20 DA FD              JSR  PRBYTE
80E8: A9 26                 LDA  #<MSG3
80EA: A0 80                 LDY  #>MSG3
80EC: 20 3A DB              JSR  STROUT
80EF: 20 0C FD              JSR  GET
80F2: C9 D9                 CMP  #$D9
80F4: F0 07                 BEQ  ZEROFIL
80F6: C9 F9                 CMP  #$F9
80F8: F0 03                 BEQ  ZEROFIL
80FA: 4C 5E 80              JMP  START ;CHANGE IF RELOCATING
80FD: A9 00        ZEROFIL  LDA  #$00
80FF: AA                    TAX
8100: E0 0B        CLEAR    CPX  #$0B
8102: F0 0A                 BEQ  PASSLINE
8104: 9D 00 02              STA  BUFF,X
8107: 9D 00 03              STA  BUFF2,X
810A: E8                    INX
810B: 18                    CLC
810C: 90 F2                 BCC  CLEAR
810E: A9 37        PASSLINE LDA  #<MSG4
8110: A0 80                 LDY  #>MSG4
8112: 20 3A DB              JSR  STROUT
8115: A9 A0                 LDA  #$A0
8117: 85 33                 STA  $33
8119: 20 6A FD              JSR  INPUT
811C: A0 02                 LDY  #>BUFF
811E: A2 00                 LDX  #<BUFF
8120: A9 03                 LDA  #>BUFF2
8122: 85 72                 STA  $72
8124: A9 00                 LDA  #<BUFF2
8126: 85 71                 STA  $71
8128: A9 0A                 LDA  #$0A
812A: 20 E2 E5              JSR  MOVSTR
812D: A9 49                 LDA  #<MSG5
812F: A0 80                 LDY  #>MSG5
8131: 20 3A DB              JSR  STROUT
8134: A9 00                 LDA  #<BUFF2
8136: A0 03                 LDY  #>BUFF2
8138: 20 3A DB              JSR  STROUT
813B: A9 26                 LDA  #<MSG3
```

```
813D: A0 80              LDY  #>MSG3
813F: 20 3A DB           JSR  STROUT
8142: 20 0C FD           JSR  GET
8145: C9 D9              CMP  #$D9
8147: F0 07              BEQ  OKAY
8149: C9 F9              CMP  #$F9
814B: F0 03              BEQ  OKAY
814D: 18                 CLC
814E: 90 AD              BCC  ZEROFIL
8150: A9 00      OKAY    LDA  #$00
8152: 85 07              STA  COUNTER
8154: AA                 TAX
8155: A8                 TAY
8156: E0 0A      CRYPT   CPX  #$0A
8158: F0 0D              BEQ  TOMB
815A: B1 05              LDA  (ADLO),Y
815C: 18                 CLC
815D: 7D 00 03           ADC  BUFF2,X
8160: 91 05              STA  (ADLO),Y
8162: E8                 INX
8163: C8                 INY
8164: 18                 CLC
8165: 90 EF              BCC  CRYPT
8167: A5 07      TOMB    LDA  COUNTER
8169: C9 14              CMP  #$14
816B: F0 07              BEQ  FINIS
816D: E6 07              INC  COUNTER
816F: A2 00              LDX  #$00
8171: 18                 CLC
8172: 90 E2              BCC  CRYPT
8174: 20 58 FC   FINIS   JSR  HOME
8177: A9 56              LDA  #<MSG6
8179: A0 80              LDY  #>MSG6
817B: 20 3A DB           JSR  STROUT
817E: 4C 03 E0           JMP  BASIC

--End assembly, 385 bytes, Errors: 0
```

what address your program loads into and how long it is, as you'll need this information when BSAVEing the resulting encrypted file. You'll definitely need to know where it is when encrypting it. You can determine these values after BLOADing your file by viewing memory locations $AA60 and $AA61... the file length... and locations $AA72 and $AA73... where the file was BLOADed to. For example, if the bytes in $AA60–1 are 23 02, and $AA72–3's contents are 00 08, then you'd

BSAVE YOURPROGRAM.SCRAM,A$0800,L$0223

It's best not to BSAVE the encrypted code under the same name as the original file. This could have disasterous consequences... especially if you forget the password you assigned to the encrypting. To be safe, always keep a backup of any files you intend to encrypt on a separate disk, in a place free from shifty eyes and light fingers.

To enact destiny, type either CALL 32768 from BASIC or 8000G from the monitor. After the title and sundry details are printed to the screen, you're asked to enter your program's starting address. The program will beep at you if you don't enter a legitimate hex address. $0800 is the default address.

You're next asked for a ten digit password. Ten digits aren't really crucial... it'll take less and pad the remainder with zeroes, and will truncate longer entries. The way the encrytion works, however, will leave your text file's bytes intact where there are zeroes in your password. Be warned.

After checking with you as to the validity of your entry, the encryption starts.

Into the Crypt

Despite the apparent simplicity of its encoding scheme, code scrambled with *Binary File Encryption* is remarkably difficult to unscramble without beforehand knowledge of the ten letter password, or key. As no clue of the password is given in either the encrytion or decryption programs, even unauthorized use of these files on your scrambled programs will yield little but frustration. This is all the more reason to ensure that you remember the password.

In brief, program two inhales your password and jams it in the $0300 area for safekeeping. It then takes your program's code, ten bytes at a time, and subtracts the hex values of your password from it. When two hundred and ten bytes have been subjected to this abuse, it ends its reign of terror and tells you to BSAVE the resultant code.

First, an explanation is in order. For the most part, two hundred and ten scrambled bytes will go a long way in confusing unwanted onlookers. If you tend to write mammoth programs or have similarly large binary data files, you may want some more protection. If this is the case, go through program one again, directing it to encrypt at an address other than the original one you chose.

More caution is advised in this case... if any area of your program gets encoded twice through unwittedly overlapping starting addresses, it'll be next to impossible to get the original bytes back. Also, all the starting addresses and corresponding passwords of the blocks you choose to encrypt must be recalled when you use program one. You'll likely have to write them down somewhere.

Intentionally overlapping (encoding a single area of code any number of times) will practically guarentee that code's security, but even more caution must be taken. If you choose to encode a block ten times, you must enter the same address for each iteration, and write down each password. When decoding, you'll need to either remember, or have handy the encoded block's address and the ten passwords. The multiple passwords *must* be entered individually and in *reverse order* when decoding. If you choose to use multiple passwords, you'll have to run the encryption and decryption programs as many times as passwords you intend to use.

There's no limit, save that of available RAM, to the number of two hundred and ten byte program blocks you can scramble, or to the number of passwords you can use in a single file. Still, common sense should be employed when possible. If you forget the password or address for *one* encrypted block, you may as well scrap the file for good.

@N.G$TYZ[rsR#)

The subhead above translates to 'Scrambled Eggs' when its characters are added to a key using my last name and a period to round out the ten character requirement. The actual key is 'RUDZINSKI.RUDZ'.

Essentially, this is what program one does. If you look at the actual decrypting code in the listing, you'll note it's similar to program two's encrypting scheme except for four bytes, where the given password's values are added to the encrypted program. Instructions for its use are similar to those of program two. BLOAD it, then BLOAD your encrypted data, making note of the first location it BLOADs into.

The prompts in program one are somewhat more curt... even vague in a way. User friendliness is fine, but a decoding program in unauthorized hands should be downright unneighbourly. Just

Listing 2, the encryptor

```
:ASM
                *
                *
                *        BINARY FILE ENCRYPTION
                *
                *          COPYRIGHT (C) 1984
                *
                *            JOHN RUDZINSKI
                *
                *
                ADLO     EQU  $05
                ADHI     EQU  $06
                COUNTER  EQU  $07
                BASIC    EQU  $E003
                BEEP     EQU  $FF3A
                BUFF     EQU  $200
                BUFF2    EQU  $300
                COUT     EQU  $FDED
                GET      EQU  $FD0C
                HOME     EQU  $FC58
                INPUT    EQU  $FD6A
                MOVSTR   EQU  $E5E2
                PRBYTE   EQU  $FDDA
                STROUT   EQU  $DB3A
                *
8000: 18                 CLC
8001: 90 71              BCC  START
                *
8003: C1 D0 D0  MSG1     ASC  "APPLE BINARY FILE ENCODER"8D8D
8006: CC C5 A0 C2 C9 CE C1 D2
800E: D9 A0 C6 C9 CC C5 A0 C5
8016: CE C3 CF C4 C5 D2 8D 8D
801E: C3 CF D0           ASC  "COPYRIGHT (C) JOHN RUDZINSKI"8D8D8D8D
8021: D9 D2 C9 C7 C8 D4 A0 A8
8029: C3 A9 A0 CA CF C8 CE A0
8031: D2 D5 C4 DA C9 CE D3 CB
8039: C9 8D 8D 8D 8D
803E: CE CF D2           ASC  "NORMAL START: $0800. CHANGE? "00
8041: CD C1 CC A0 D3 D4 C1 D2
8049: D4 BA A0 A4 B0 B8 B0 B0
8051: AE A0 C3 C8 C1 CE C7 C5
8059: BF A0 00
805C: 8D 8D     MSG2     HEX  8D8D
805E: C5 CE D4           ASC  "ENTER START ADDRESS: "00
8061: C5 D2 A0 D3 D4 C1 D2 D4
8069: A0 C1 C4 C4 D2 C5 D3 D3
8071: BA A0 00
                *
8074: 20 58 FC  START    JSR  HOME
8077: A9 08              LDA  #$08        ;DEFAULT
8079: 85 06              STA  ADHI
807B: A9 00              LDA  #$00
807D: 85 05              STA  ADLO        ;DITTO
807F: A9 A4              LDA  #$A4        ;$
8081: 85 33              STA  $33         ;PROMPT LOCATION
8083: A9 03              LDA  #<MSG1      ;LOW ADDRESS
8085: A0 80              LDY  #>MSG1      ;HIGH ADDRESS
8087: 20 3A DB           JSR  STROUT      ;APPLESOFT STRING PRINT
808A: 20 0C FD           JSR  GET
                *
808D: C9 D9              CMP  #$D9        ;Y
808F: F0 07              BEQ  ADDENDA
8091: C9 F9              CMP  #$F9        ;y
8093: F0 03              BEQ  ADDENDA
8095: 18                 CLC
8096: 90 37              BCC  GETPASS
8098: A9 5C     ADDENDA  LDA  #<MSG2
809A: A0 80              LDY  #>MSG2
809C: 20 3A DB           JSR  STROUT
809F: 20 6A FD           JSR  INPUT
80A2: A2 00              LDX  #$00
```

Apple Encryption

```
80A4: E0 04           GETBYTE CPX  #$04
80A6: F0 79                   BEQ  ENSURE
80A8: BD 00 02                LDA  BUFF,X
80AB: C9 B0                   CMP  #$B0      ;0
80AD: 30 1A                   BMI  ERROR
80AF: C9 BA                   CMP  #$BA      ;9(+1)
80B1: 10 07                   BPL  HEXLTR
80B3: 9D 00 03                STA  BUFF2,X
80B6: E8                      INX
80B7: 18                      CLC
80B8: 90 EA                   BCC  GETBYTE
80BA: C9 C1           HEXLTR  CMP  #$C1
80BC: 30 0B                   BMI  ERROR
80BE: C9 C7                   CMP  #$C7      ;F(+1)
80C0: 10 07                   BPL  ERROR
80C2: 9D 00 03                STA  BUFF2,X
80C5: E8                      INX
80C6: 18                      CLC
80C7: 90 DB                   BCC  GETBYTE
80C9: 20 3A FF        ERROR   JSR  BEEP
80CC: 18                      CLC
80CD: 90 C9                   BCC  ADDENDA
80CF: 18              GETPASS CLC
80D0: 90 52                   BCC  PASSINP
                      *
80D2: 8D 8D           MSG3    HEX  8D8D
80D4: A4 B0 B8                ASC  "$0800 ASSUMED."8D8D
80D7: B0 B0 A0 C1 D3 D3 D5 CD
80DF: C5 C4 AE 8D 8D
80E4: C5 CE D4        MSG4    ASC  "ENTER 10 DIGIT PASSWORD:"00
80E7: C5 D2 A0 B1 B0 A0 C4 C9
80EF: C7 C9 D4 A0 D0 C1 D3 D3
80F7: D7 CF D2 C4 BA 00
80FD: 8D 8D           MSG5    HEX  8D8D
80FF: D9 CF D5                ASC  "YOU ENTERED: $"00
8102: A0 C5 CE D4 C5 D2 C5 C4
810A: BA A0 A4 00
810E: 8D              MSG6    HEX  8D
810F: C9 D3 A0                ASC  "IS THIS CORRECT? "00
8112: D4 C8 C9 D3 A0 C3 CF D2
811A: D2 C5 C3 D4 BF A0 00
                      *
8121: 18              ENSURE  CLC
8122: 90 0A                   BCC  PMSG5
8124: A9 D2           PASSINP LDA  #<MSG3
8126: A0 80                   LDY  #>MSG3
8128: 20 3A DB                JSR  STROUT
812B: 18                      CLC
812C: 90 75                   BCC  PASS2
                      *
812E: A9 FD           PMSG5   LDA  #<MSG5
8130: A0 80                   LDY  #>MSG5
8132: 20 3A DB                JSR  STROUT
8135: A2 00                   LDX  #$00
8137: E0 04           LOGIC   CPX  #$04
8139: F0 23                   BEQ  EEYORE
813B: BD 00 03                LDA  BUFF2,X
813E: C9 BA                   CMP  #$BA      ;9(+1)
8140: 10 17                   BPL  LTR
8142: E9 AF                   SBC  #$AF
8144: E0 00           COMPARE CPX  #$00
8146: F0 0B                   BEQ  SHIFT
8148: E0 02                   CPX  #$02
814A: F0 07                   BEQ  SHIFT
814C: 9D 00 03        STORE   STA  BUFF2,X
814F: E8                      INX
8150: 18                      CLC
8151: 90 E4                   BCC  LOGIC
8153: 20 53 82        SHIFT   JSR  LEFT
8156: 18                      CLC
8157: 90 F3                   BCC  STORE
8159: E9 B7           LTR     SBC  #$B7
815B: 18                      CLC
815C: 90 E6                   BCC  COMPARE
                      *
815E: AD 00 03        EEYORE  LDA  BUFF2
8161: 4D 01 03                EOR  BUFF2+1
8164: 48                      PHA
8165: AD 02 03                LDA  BUFF2+2
8168: 4D 03 03                EOR  BUFF2+3
816B: 8D 00 03                STA  BUFF2
816E: 68                      PLA
816F: 8D 01 03                STA  BUFF2+1   ;LE VOILA
8172: 85 06                   STA  ADHI
8174: 20 DA FD                JSR  PRBYTE    ;PRINT THAT BYTE
8177: AD 00 03                LDA  BUFF2
817A: 85 05                   STA  ADLO
817C: 20 DA FD                JSR  PRBYTE    ;AN' THAT ONE, TOO
                      *
817F: A9 0E                   LDA  #<MSG6
8181: A0 81                   LDY  #>MSG6
8183: 20 3A DB                JSR  STROUT
8186: 20 0C FD                JSR  GET
8189: C9 D9                   CMP  #$D9      ;Y
818B: F0 07                   BEQ  PMSG4
818D: C9 F9                   CMP  #$F9      ;y
818F: F0 03                   BEQ  PMSG4
8191: 4C 74 80                JMP  START     ;CHANGE IF RELOCATING
                      *
8194: A9 8D           PMSG4   LDA  #$8D
8196: 20 ED FD                JSR  COUT
8199: 20 ED FD                JSR  COUT
819C: A9 E4                   LDA  #<MSG4
819E: A0 80                   LDY  #>MSG4
81A0: 20 3A DB                JSR  STROUT
81A3: A9 A0           PASS2   LDA  #$A0
81A5: 85 33                   STA  $33
81A7: A9 00                   LDA  #$00
81A9: AA                      TAX
81AA: E0 0B           CLEAR   CPX  #$0B
81AC: F0 0A                   BEQ  MOVEIT
81AE: 9D 00 02                STA  BUFF,X    ;CLEAR THE
81B1: 9D 00 03                STA  BUFF2,X   ;BUFFERS
81B4: E8                      INX
81B5: 18                      CLC
81B6: 90 F2                   BCC  CLEAR
81B8: 20 6A FD        MOVEIT  JSR  INPUT     ;GET PASSWORD
81BB: A0 02                   LDY  #>BUFF
81BD: A2 00                   LDX  #<BUFF
81BF: A9 03                   LDA  #>BUFF2
81C1: 85 72                   STA  $72
81C3: A9 00                   LDA  #<BUFF2
81C5: 85 71                   STA  $71
81C7: A9 0A                   LDA  #$0A      ;PASSWORD LENGTH
81C9: 20 E2 E5                JSR  MOVSTR    ;MOVE PASSWORD TO BUFF2
81CC: 18                      CLC
81CD: 90 30                   BCC  VERIFY
                      *
81CF: 8D 8D           MSG7    HEX  8D8D
81D1: D0 C1 D3                ASC  "PASSWORD: "00
81D4: D3 D7 CF D2 C4 BA A0 00
81DC: C4 CF CE        MSG8    ASC  "DONE."8D8D
81DF: C5 AE 8D 8D
81E3: C2 D3 C1                ASC  "BSAVE YOUR PROGRAM NOW."8D8D00
81E6: D6 C5 A0 D9 CF D5 D2 A0
81EE: D0 D2 CF C7 D2 C1 CD A0
81F6: CE CF D7 AE 8D 8D 00
                      *
81FD: 90 95           PM1     BCC  PMSG4
81FF: A9 CF           VERIFY  LDA  #<MSG7
8201: A0 81                   LDY  #>MSG7
8203: 20 3A DB                JSR  STROUT
8206: A9 00                   LDA  #<BUFF2
8208: A0 03                   LDY  #>BUFF2
820A: 20 3A DB                JSR  STROUT
820D: A9 0E                   LDA  #<MSG6
820F: A0 81                   LDY  #>MSG6
8211: 20 3A DB                JSR  STROUT
8214: 20 0C FD                JSR  GET
8217: C9 D9                   CMP  #$D9
8219: F0 07                   BEQ  PASSOK
821B: C9 F9                   CMP  #$F9
821D: F0 03                   BEQ  PASSOK
821F: 18                      CLC
8220: 90 DB                   BCC  PM1
                      *
8222: 20 58 FC        PASSOK  JSR  HOME
8225: A9 00                   LDA  #$00
8227: 85 07                   STA  COUNTER
8229: AA                      TAX
822A: A8                      TAY
822B: E0 0A           ENCRYPT CPX  #$0A
822D: F0 0D                   BEQ  ENTOMB
822F: B1 05                   LDA  (ADLO),Y
8231: 38                      SEC
8232: FD 00 03                SBC  BUFF2,X
8235: 91 05                   STA  (ADLO),Y
8237: E8                      INX
8238: C8                      INY
8239: 18                      CLC
823A: 90 EF                   BCC  ENCRYPT
823C: A5 07           ENTOMB  LDA  COUNTER
823E: C9 14                   CMP  #$14      ;20x10=200
8240: F0 07                   BEQ  FINIS
```

[redacted]

```
8242: E6 07              INC   COUNTER
8244: A2 00              LDX   #$00
8246: 18                 CLC
8247: 90 E2              BCC   ENCRYPT
                    *
8249: A9 DC        FINIS LDA   #<MSG8
824B: A0 81              LDY   #>MSG8
824D: 20 3A DB           JSR   STROUT
8250: 4C 03 E0           JMP   BASIC
                    *
8253: 0A           LEFT  ASL
8254: 0A                 ASL
8255: 0A                 ASL
8256: 0A                 ASL
8257: 60                 RTS

--End assembly, 600 bytes, Errors: 0
```

answer the prompts with the same information you used to encrypt the file in memory. If you encrypted any single block with more than one password, remember that password orders must be reversed, so that, say, password #3 is entered before password #2. The program will ask for confirmation after each input before rearranging your RAM.

Note that both programs one and two are destructive... they write to the program RAM. Unless you have multitudes of foreign spies battering on your door, take some time to practice using these programs on well backed up files. When you've an important file you need encrypted, back it up and test a number of passwords on it. Some combinations of characters produce better encryptions than others.

Decode Book

The extent to which you are willing to go to protect your data will most likely be proportional to either the sensitivity of said bytes of the magnitude of your paranoia. However, be warned... paranoids have enemies too.

There are a few things to consider about using this software. The first is that any encryption process offers the possibility of garbaging your data. You might just forget your password. There might still be an unsquashed bug in there. Your computer might glitch without your knowing it.

If your files are irreplaceable, keep backup copies... unencrypted... somewhere behind a few dozen stout locks.

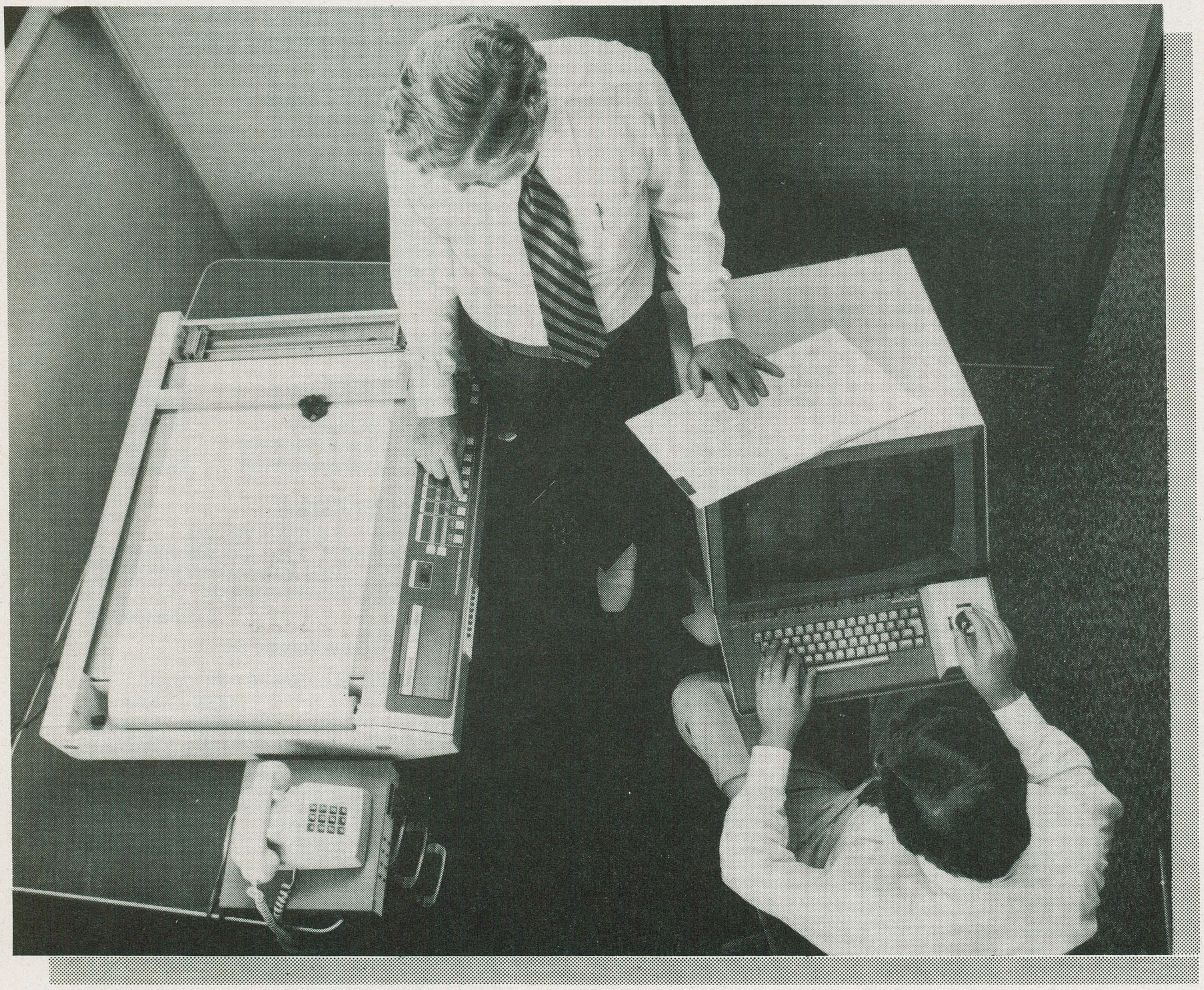
Secondly, be aware that *any* encrypted file can be cracked by someone determined enough to do so. A computer can try all the permutations of a ten letter password... admittedly, not quickly, but in a reasonable time if a clever enough routine is written to try the most likely permutations first. Most types of files, be they text files, code or data files from a spreadsheet, have characteristic patterns of bytes which a cracking program can look for to know when it has picked the right decoding string.

However, this routine, if implemented sensibly, should give you a secure, uncomplicated way to deny others access to your data. It's just the thing if you suspect that RCMP agents from higher order dimensions are sneaking into your basement and using your Ultima cheat programs without a warrant. It will also fox the wife if you fear assaults on your checkbook overlays.

In a few rare cases it can even be used to clarify certain texts. Consider running transcripts of the proceedings of the House of Commons through it...

Of course, one would only need the encryptor for that. **CN!**

PreBAS



If you want to write really tight structured code and you insist on using BASIC you should try PreBAS. It will give you a whole new perspective on programming and, what's more important, inasmuch as you already have this magazine, it's virtually free.

by Steve Rimmer

Computer languages are almost as varied as the machines that are available to run them. Once you decide to step outside the tranquil world of BASIC you can immediately become enmired in Pascal, C, Ada, Forth, Fortran... the list is inexhaustible. This is more than you can say for most programmers, who usually get exhausted fairly early on.

The unfortunate aspect of so many of these fascinating developments of the human mind is that they are frightfully, tediously, mind grindingly expensive. Furthermore, many are really hard to work with, to the extent that many users are a bit reluctant to pop for a sharp new language package that they'll very possibly never be able to use.

Now, almost any means of expressing your wishes to a computer will have some advantages over that of BASIC. A typical Microsoft BASIC package trades everything off for convenience. One of the most insidious losses in this respect is the ability to structure your programming. You may think you're writing tight code but it's just an illusion... nothing with line numbers can ever match the flexibility of compiled programming.

For all these reasons and some which are only spoken of in the dripping catacombs of the nether trolls there has come to pass...

PreBAS. Far from being a new language, PreBAS is a new way to use an existing one. PreBAS is a very simple BASIC pre-processor.

PreBAS won't make BASIC any faster or smarter or even any less gross. However, it will allow you to write large BASIC programs in a much more convenient way.

Back to Pre-BASICs

In order to use PreBAS you will need a disk BASIC package and a word processor which can create program source files. This version was intended for use with Microsoft MBASIC and WordStar in the N mode, although there are plenty of other permutations which will work. Except for some of the tricky screen formatting in the PreBAS program itself... which can be done away with... this thing will work on any system.

It still may be a bit muddy as to exactly what PreBAS does. Consider programs one and two. They're both trivial, and aren't very clever as programs go. However, this is irrelevant, as they're examples, and examples are always trivial.

Program one is written in a neat new, totally non-existent language called unBASIC. While it features BASIC keywords and syntax, it lacks line numbers. Instead, it is full of labels. One does not say GOSUB 100... one says GOSUB TYPE. This has a number of definite advantages.

The immediately useful thing about unBASIC is that it is self documenting... well, to an extent, anyway. It's a lot more revealing to say GOSUB TYPE, as opposed to GOSUB 100. Furthermore, when you're cheerfully programming away you don't have to keep track of the line numbers of your utility subroutines... just give 'em easy to remember names.

UnBASIC allows for names that are up to twenty-eight characters in length. The first character must be alpha, there can be no spaces and the last character in the label must be a colon. This is the usual convention with most structured languages. If you would like to have spaces in long labels, use the underscore character.

Now, unBASIC can't be created as you would a real BASIC program. You can't type it into Microsoft BASIC because each line, being numberless, would be treated by BASIC as a direct command and not stored. The labels would completely fox the interpreter. Thus, unBASIC programs are done, as is the case with real world compilers, using a word processor or text editor to create a program text file. The file is then run through PreBAS.

I know... we still haven't gotten to exactly what PreBAS is. Well, when you run PreBAS it will inhale program one and spit out program two. Program two, while a bit messy, is a legitimate BASIC file. Invoke MBASIC, LOAD the program and run it and it will be quite happy doing so... unless there are programming errors in the code itself.

While writing in the style of unBASIC will take some getting used to you'll find that it is ultimately a lot easier when you're working with fairly huge programs. A word processor makes a much more powerful editor than the one in BASIC and being able to reference labels is really fine experience.

Program 1

This is an unBASIC program

```
         'SAMPLE STRUCTURED PROGRAM
         'TO BE PROCESSED WITH
         'WITH PreBAS AND RUN
         'ON MICROSOFT BASIC
START:
         GOSUB WOMBAT

         CRLF$ = CHR$(13) + CHR$(10)
         A$ = CRLF$ + "What is your name"
         GOSUB TYPE
         GOSUB GETANSWER
         N$ = A$

         A$ = CRLF$ + "What is your quest"
         GOSUB TYPE
         GOSUB GETANSWER
         Q$ = A$

         A$ = CRLF$ + "What is the air /
           speed velocity of an /
           unladen swallow"

         GOSUB TYPE
         GOSUB GETANSWER
         S$ = A$

         GOSUB DISKIT
         END

         'SUBROUTINES

TYPE:
         FOR X=1 TO LEN(A$)
         PRINT MID$(A$,X,1);
         NEXT X
         RETURN

GETANSWER:
         INPUT A$
         RETURN

DISKIT:
         ON ERROR GOTO DISKERR
         OPEN "I",1,"GRAIL.DOC"
         INPUT #1,N$,Q$,S$
         CLOSE 1
         A$ = CRLF$ +  "The last man to /
           come this way was " + N$
         GOSUB TYPE
         A$ = CRLF$ +  "He quested for "
         + Q$ GOSUB TYPE
         A$ = CRLF$ +  "He thought that
         the air speed velocity of
         an unladen swallow was " +  S$
         GOSUB TYPE
         A$ = CRLF$ +  "He never returned"
         GOSUB TYPE

DISKIT1:
         OPEN "O",1,"GRAIL.DOC"
         PRINT #1,N$,Q$,S$
         CLOSE 1
         RETURN

DISKERR:
         A$ = CRLF$ + "You be the first
           man to come this way..."
         GOSUB TYPE
         ON ERROR GOTO 0
         RESUME DISKIT1

%END OF CODE
```

Program 2

This is a real BASIC program

```
20      'SAMPLE STRUCTURED PROGRAM    '
30      'TO BE PROCESSED WITH    '
40      'WITH PreBAS AND RUN    '
50      'ON MICROSOFT BASIC    '
60  REM START:  '
70  REM   '
80  ' GOSUB 65535  ...ERROR! Undefined
line. SYMBOL WOMBAT
90  REM   '
100     CRLF$ = CHR$(13) + CHR$(10)    '
110     A$ = CRLF$ + "What is your name"
120     GOSUB 300   '
130     GOSUB 360   '
140     N$ = A$    '
150  REM   '
160     A$ = CRLF$ + "What is your quest"
170     GOSUB 300   '
180     GOSUB 360   '
190     Q$ = A$    '
200  REM   '
210     A$ = CRLF$ + "What is the air
   speed velocity of an unladen
   swallow"   '
220     GOSUB 300   '
230     GOSUB 360   '
240     S$ = A$    '
250  REM   '
260     GOSUB 400    '
270     END   '
280  REM   '
290     'SUBROUTINES    '
300  REM            '
310  REM TYPE:  '
320     FOR X=1 TO LEN(A$)     '
330     PRINT MID$(A$,X,1);     '
340     NEXT X    '
350     RETURN    '
360  REM   '
370  REM GETANSWER:   '
380     INPUT A$    '
390     RETURN    '
400  REM   '
410  REM DISKIT:   '
420     ON ERROR GOTO 600    '
430     OPEN "I",1,"GRAIL.DOC"     '
440     INPUT #1,N$,Q$,S$     '
450     CLOSE 1    '
460     A$ = CRLF$ +  "The last man to
   come this way was " + N$     '
470     GOSUB 300    '
480     A$ = CRLF$ +  "He quested for "
   + Q$    '
490     GOSUB 300    '
500     A$ = CRLF$ +  "He thought that
   the air speed velocity of an
```

```
    unladen swallow was " +  S$    '
510    GOSUB 300   '
520    A$ = CRLF$ +  "He never
    returned"    '
530    GOSUB 300    '
540    REM    '
550    REM DISKIT1:   '
560    OPEN "O",1,"GRAIL.DOC"     '
570    PRINT #1,N$,Q$,S$      '
580    CLOSE 1     '
590    RETURN    '
600    REM     '
610    REM DISKERR:    '
620    A$ = CRLF$ + "You be the first
    man to come this way..."     '
630    GOSUB 300    '
640    ON ERROR GOTO 0     '
650    RESUME 540     '
660    REM    '
670    REM %END OF CODE
```

Program 3

This is the symbol table for Program 1

50	1	START:
300	2	TYPE:
360	3	GETANSWER:
400	4	DISKIT:
540	5	DISKIT1:
600	6	DISKERR:

Program 4

```
10 '    PreBAS BASIC PRECOMPILER
20 '    FOR USE WITH MICROSOFT BASIC
30 '    COPYRIGHT (c) 1984 STEVE RIMMER
40 '
50 '     ++++ DEFINES ++++
60 CRLF$ = CHR$(13) + CHR$(10)
70 CLS$ = CHR$(26)
80 HM$ = CHR$(30)
90 LF$ = CHR$(10)
100 PD = 20
110 DIS$ = CLS$ + STRING$(10,LF$) +
    SPACE$(PD)
120 ALPHA$ = "ABCDEFGHIJKLMNOP
    QRSTUVWXYZ"
130 LINC = 10  'LINE NUMBER INCREMENT
140 DIM ARR$(255,8)
150 '    ++++ RUNNING CODE ++++
160 PRINT DIS$ "What file do you want
    processed";
170 INPUT FILE$
180 IF FILE$ = "" THEN PRINT CLS$ : END
190 IF LEN(FILE$) > 10 THEN ER=1 : GOSUB
    640 : GOTO 160
200 IF INSTR(FILE$,".") <> 0 THEN ER=2 :
    GOSUB 640 : GOTO 160
210 '--OPEN FILE AND LOOK FOR SYMBOLS
220 PRINT DIS$ "First pass through "
    FILE$ ". Assembling symbols."
    LF$ LF$
230 ON ERROR GOTO 910
240 OPEN "I",1,FILE$ + ".SRC"
250 ON ERROR GOTO 0
260 OPEN "O",2,FILE$ + ".SYM"
270 LNUM = LINC
280 SYM = 1
290 EFLAG = 0
300 PNT = 1
310 ARR$(0,0) = STRING$(2,255) +
    "@@@@:" + STRING$(22," ")
320 IF EOF(1) THEN 370
330  LINE INPUT #1,A$
340  GOSUB 760 'COMPILE SYMBOLS
350 IF ER=4 THEN 370
360 GOTO 320
370 CLOSE
380 IF ER = 4 THEN GOSUB 640 : GOTO 540
390 '--SECOND PASS
400 PRINT DIS$ "Second pass through "
    FILE$ ". Replacing symbols."
    LF$ LF$
410 A$ = ""
420 LNUM = LINC
430 OPEN "I",1,FILE$ + ".SRC"
440 OPEN "O",2,FILE$ + ".BAS"
450 IF EOF(1) THEN 490
460  LINE INPUT #1,A$
470  GOSUB 960 'FIND THE SYMBOLS
480 GOTO 450
490 CLOSE
500 '--SAY WAIT
510 PRINT CRLF$ TAB(PD) "[Hit any key to
    continue.]" ;
520 C$ = INPUT$(1)
530 PRINT HM$
540 '--REPORT STATUS
550 PRINT DIS$ "Source file " FILE$ "
    PreBAS pre-compilation complete."
560 PRINT TAB(PD)"Lines of code        "
    INT((LNUM-1) / LINC)
570 PRINT TAB(PD)"Errors               "
    EFLAG
580 PRINT TAB(PD)"Symbols              "
    SYM
590 PRINT TAB(PD)"Symbol space left    "
    (2048 -PNT)
600 ER = 0
610 GOSUB 640
620 PRINT CLS$
630 END
640 ' --ERROR TRAPPER (NON DISK)
650 PRINT HM$ STRING$(20,LF$) TAB(PD);
660 IF ER>0 THEN PRINT "+++++ Error !"
670 IF ER = 0 THEN PRINT
680 IF ER=1 THEN PRINT "That's not a
    valid file name."
690 IF ER=2 THEN PRINT "Do not add the
    file extension."
700 IF ER=3 THEN PRINT "Cannot open
    source file."
710 IF ER=4 THEN PRINT "Symbol table
    space exhausted."
720 PRINT TAB(PD) "[Hit any key to
    continue.];
730 C$ = INPUT$(1)
740 PRINT CLS$;
750 RETURN
760 '--SORT OUT SYMBOLS FROM DISK FILE
770 IF A$ = "" THEN 890
780 IF INSTR(ALPHA$,LEFT$(A$,1)) = 0
    THEN 890
790 IF INSTR(A$,":") = 0 THEN PRINT A$ "
    <===== SYMBOL ERROR." : EFLAG =
    EFLAG + 1 : GOTO 890
800 SYM = SYM + 1
810 N$ = SPACE$(32)
820 J=INT(LNUM/256) : K = LNUM - 256 * J
830 LSET N$ = CHR$(K) + CHR$(J) + A$
840 PRINT LNUM TAB(10) PNT TAB(20) A$
850 PRINT #2, STR$(LNUM) TAB(10)
    STR$(PNT) TAB(20) A$
860 K = INT(PNT/256) : J = PNT - K
870 ARR$(J,K) = N$
880 PNT = PNT + 1
890 LNUM = LNUM + LINC
900 RETURN
910 '--ERROR TRAPPER (DISK)
920 CLOSE
930 ER = 3
940 GOSUB 640
950 RESUME 160
960 '--LOCATE SYMBOLS IN STATEMENTS
970 SFLAG = 0
980 LFLAG = 0
990 IF LEFT$(A$,1) <> " " AND
    LEFT$(A$,1) <> CHR$(9) THEN 1170
1000 A$ = A$ + " "
1010 IF INSTR(A$,"GOTO") <> 0 AND
     INSTR(A$,"GOTO 0") = 0 THEN SFLAG =
     1 : CS = INSTR(A$,"GOTO") + 5 :
     CSYM$ = MID$(A$,CS,(INSTR(CS+1,
     A$," ")))
1020 IF INSTR(A$,"GOSUB") <> 0 THEN
     SFLAG = 1 : CS = INSTR(A$,"GOSUB")
     + 6 : CSYM$ = MID$(A$,CS,
     (INSTR(CS+1,A$," ")))
1030 IF INSTR(A$,"RESUME") <> 0 THEN
     SFLAG = 1 : CS = INSTR(A$,"RESUME")
     + 7 : CSYM$ = MID$(A$,CS,
     (INSTR(CS+1,A$," ")))
1040 IF SFLAG = 0 THEN 1170
1050 IF RIGHT$(CSYM$,1) = " " THEN CSYM$
     = LEFT$(CSYM$,LEN(CSYM$)-1) : GOTO
     1050
1060 CSYM$ = CSYM$ + ":"
1070 '--LOOK FOR SYMBOL IN SYMBOL TABLE
1080 X=PNT-1 : J=0 : K=0
1090 WHILE MID$(ARR$(J,K),3,
     (LEN(CSYM$))) <> CSYM$ AND X >-1
1100 K=INT(X/256) : J=X-K
1110 LFLAG = ASC(LEFT$(ARR$(J,K),1)) +
     256 * ASC(MID$(ARR$(J,K),2,1))
1120 X=X-1
1130 WEND
1140 CSYM$ = LEFT$(CSYM$,
     INSTR(CSYM$,":")-1)
1150 '--REPLACE SYMBOL WITH LINE NUMBER
1160 A$ = LEFT$(A$,INSTR(A$,CSYM$)-2) +
     STR$(LFLAG)
1170 '--DISPLAY LINE AND PUT IT IN FILE
1180 IF INSTR(ALPHA$,LEFT$(A$,1))<>0 OR
     A$ = " " OR LEFT$(A$,1) = "%" THEN
     A$ = "REM " + A$
1190 IF LEN(A$) < 3 THEN A$ = "REM " +
     A$
1200 IF LFLAG = 65535! THEN A$ = "'" +
     A$ + "   ...ERROR! Undefined line.
     SYMBOL "  + CSYM$ : EFLAG =
     EFLAG + 1
1210 IF INSTR(ALPHA$,RIGHT$(A$,1))=0
     THEN A$ = A$ + "     '"
1220 PRINT LNUM "  " A$;
1230 IF SFLAG = 1 THEN PRINT TAB(45)
     "Symbol " CSYM$ " " LFLAG ELSE
     PRINT
1240 LNUM = LNUM + LINC
1250 PRINT #2,LNUM " " A$
1260 RETURN
```

Speakings

Obviously, unBASIC has the same syntax as real BASIC... unBASIC will support whatever instructions your real BASIC supports. However, there are some limitations which I've imposed on unBASIC to keep the PreBAS program short and uncomplicated.

To begin with, you can only have one label in a line, this being at the end. Thus

IF A=1 THEN GOTO ERROR

is a legal line, while

IF A=1 THEN GOTO ERROR ELSE GOTO NOERROR

is not. Notice to that you have to have a GOTO after THEN in an IF THEN statement. BASIC doesn't care whether it's there or not, but PreBAS needs it to locate the label after it.

The unBASIC file can be called whatever you like but it must have the extension .SRC, for "source".

Each line in the file will be parsed, or split into its component parts, separately by PreBAS. The first character in a line should be a tab character, CHR$(9) or control I, unless that line is a label, in which case there has to be a colon at the end.

There are a few things that will confuse PreBAS in this respect. For example,

PRINT "Type your name:

is a legal, though sloppy, line of BASIC programming. It may upset the pre-compiler, however.

An unBASIC line can't be longer than two hundred and fifty five characters, although I can't imagine why it would want to be. You should avoid referencing non-existent labels, as these will generate a non-fatal error in your BASIC file in which the line with the bad label is REM'd out and points to an impossibly high line number.

You should also refrain from using colons in references to labels. That is, if you want to say "GOSUB WOMBAT" don't say "GOSUB WOMBAT:"... the system may think that "GOSUB WOMBAT:" is the label that starts a subroutine.

The function of PreBAS is fairly simple. It is a two pass pre-compiler. That is, it reads the unBASIC file through twice. The first shot is to assemble a symbol table. It looks through the file and spots any lines which have valid labels with colons after them. It will stuff these in the array ARR$(J,K), along with their line numbers.

Ahah... there are no line numbers! Well, there will be in a while. At the moment, though, we have virtual line numbers. The first line in the SRC file is line ten, the next is line twenty and so on.

The first pass of the compiler also produces a symbol listing file, which will have the extension .SYM. You may find this useful in debugging your eventual real BASIC program if you print it out... otherwise, it can be scratched.

Having completed the first pass PreBAS will open the SRC file again and read in the first line. Assuming it isn't a REM... first lines often are... it will scan it for the occurrence of GOTO, GOSUB and RESUME, the only BASIC keywords which can reference labels. It ignores GOTO 0, as this is a constant used to disable an ON ERROR statement. If you want GOTO 0 to appear in your BASIC program you'll have to cheat and write it into your unBASIC SRC file.

Whenever PreBAS locates one of these keywords it will parse out the symbol following it and scan its symbol table for that string. Assuming the symbol is in the table it will extract the line number from the array and replace the symbol with the line number.

An unlocatable symbol will be replaced with the number 65536 and the line REM'd out.

The program also puts a REM symbol at the end of each line. This is because BASIC gets upset if you try to load a program which has any lines that have no alpha characters in them... which can happen if you leave a blank line in your unBASIC file.

Striped BAS

While using your word processor to create small programs may seem a bit pointless, this system really gets useful when you start working on complex applications. The resulting BASIC files can be debugged and modified just as you would a normal program, but you'll find that you write much less freaky code using PreBAS.

Finally, of course, writing in unBASIC will introduce you to the concepts of structured programming which are prevalent in most other languages. While it's still emminently possible to write spaghetti code with PreBAS it's a lot easier not to.

PreBAS is one of those things which you probably aren't aware of a burning need for just now. However, if you allow its incredible karmic splendor to envelop you, your programming will never be the same. It will be readable, logical and neat. There will be REM marks everywhere, but that doesn't matter.

Most important, you'll have another new toy to confuse the family with and something new to print out to use to paddle the dog. Technology... what a trip. **CN!**

There Are Other Ways To Get CN!

Getting into your Kodiak foot sarcophogia and your goose filled parka . . . be sure to stun the geese first . . . and trudging out across the frozen tundra to the newsstand for your copy of Computing Now! may seem like a drag. For one thing, you have to trudge all the way back to your shack before you can check out the latest news, applications, software and equipment reviews.

We'd like to point out that there are other ways to get your magazine. Consider the Amateur Radio League of Ripped Brain, Saskatchewan. For seven bucks an issue they'll transmit the text of CN! to you in morse code . . . as soon as they get their key working. If you aren't into atmospheric technology, you might want to try Max's Aquatic Transport Service, which will send small parcels to your door in sealed containers by water. The plumbing hookup's a hassle, mind. Finally, there is Flying Fellissi Brothers Transport, our favourite, which has this huge supply of arrows ...

If you don't like any of the above plans, you may be interested to know that we have come to another arrangement with the post office. We give them money, and they move magazines for us. In fact, they are quite prepared to freeze on your behalf to get you your issue, which is more than either of the Fellissi brothers is up for.

If you want to watch the post office freeze, send $22.95 for a one year subscription to CN!, or $37.95 for years to:

Computing Now! Subscriptions
25 Overlea Boulevard,
Suite 601
Toronto, Ontario
M4H 1B1

Moorshead Publications
ORDER FORM

Subscriptions:

Please complete reverse side of order form to start or renew a subscription.

Back Issues: $4.00 each plus 7% Ontario P.S.T.
Please circle issues desired.

1983	April	May	June	July	August
	September	October	November		December
1984	January	February	March	April	May
	June	July	August	Sept.	Oct.

On the following items please add $1.00 for postage and handling plus 7% Ontario provincial sales tax.

Special Publications:

ITEM	QTY	AMOUNT
Hobby Projects $3.95	......	$ ....
Electronic Circuit Design $3.95	......	$ ....
Projects Book No. 2 $3.95	......	$ ....
Personal Computer Guide $3.95	......	$ ....
50 Top Projects $4.95	......	$ ....
Your First Computer $3.95	......	$ ....
Computers in Small Business $3.95	......	$ ....

Binders:

Imprinted ☐ Electronics Today; ☐ Computing Now!
☐ Moorshead Publications $9.95 each plus 7% P.S.T.

BOOKSHELF ORDER FORM

Code	Title	Price
e.g. BP 12	(Short form is OK)	
....	....	$..........
....	....	$..........
....	....	$..........
....	....	$..........
....	....	$..........
....	....	$..........

SOFTWARE ORDER FORM

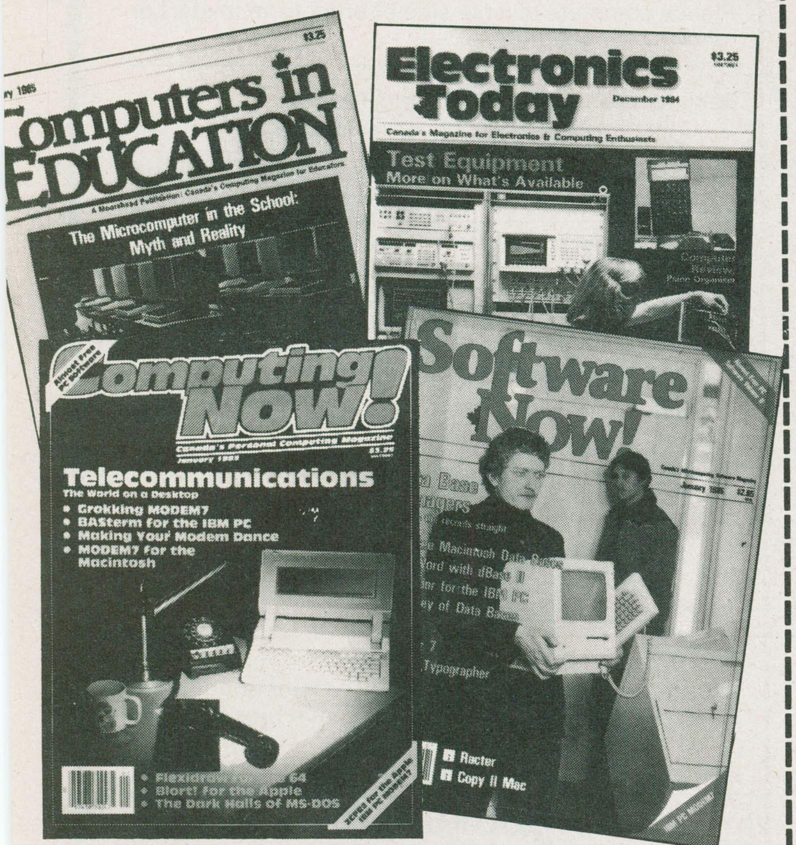
Title	Price
....	$..........
....	$..........
....	$..........
....	$..........
....	$..........
....	$..........

Sub Total $ ....
Tax (Ontario Residents) $ ....
Postage $ ....
Total Enclosed $ ....

Orders from the Bookshelf are tax exempt. Please add $1.00 for postage. Remember to put your name and address on reverse side. See over for mailing details.

Do you currently subscribe to Electronics Today	Yes ☐ No ☐
Do you currently subscribe to Computing Now!	Yes ☐ No ☐
Do you currently subscribe to Computers in Education	Yes ☐ No ☐
Do you currently subscribe to Software Now!	Yes ☐ No ☐

BE SURE OF YOUR ISSUE EACH MONTH. SUBSCRIBE TODAY.



BOOKS, BACK ISSUES, SPECIAL PUBLICATIONS, BINDERS — SEE OVER

Moorshead Publications

Suite 601, Overlea Blvd., Toronto, Ontario M4H 1B1.

MERCHANDISE ORDER ☐ Please fill out form overleaf

SUBSCRIPTIONS: ☐ NEW SUBSCRIPTION ☐ RENEWAL

Electronics Today
☐ One year (12 issues) **$19.95** ☐ Two years (24 issues) **$34.95.**

Computing Now!
☐ One year (12 issues) **$22.95** ☐ Two years (24 issues) **$37.95**

Computers in Education
☐ One year (10 issues) **$25.00** ☐ Two years (20 issues) **$45.00**

Software Now!
☐ One year (12 issues) **$19.95** ☐ Two years (24 issues) **$34.95**

For U.S. please add $3.00 per year ☐ other countries add $5 per year ☐

NAME ____________________

ADDRESS ____________________

TOWN/CITY ____________ PROVINCE/STATE ____________

CODE ____________ DATE ____________

POSTAL CODE ____________

☐ Cheque enclosed DO NOT send cash

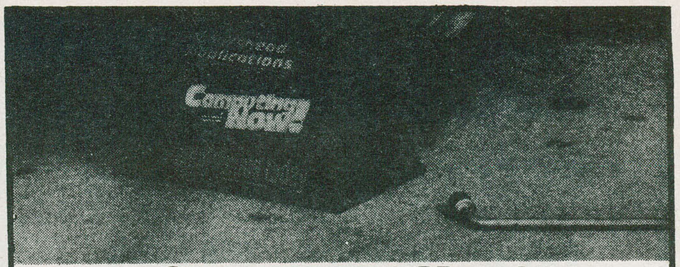
☐ Mastercard Account No. ____________

☐ Visa Account No. ____________

☐ American Express Account No. ____________

Expiry Date ____________

Signature ____________



Computing Now! Car Ramp/Magazine Binder

It's spring again, and time to change the oil and find out what is causing that annoying rattle when the engine's idling. We know what a drag it can be, especially when your head gets stuck between the ground and the oil pan, so we are offering our incredible new Computing ow! CAR RAMPS!

Covered in black vinyl, these amazing devices not only support up to 1 kg (2.2 lb.), they also happen to hold 12 issues worth of our magazines: Electronics Today, Computing Now!, and Special editions from Moorshead Publications. The world's top inventors were hired at enormous expense to design the spring gadget that holds the magazines without cutting or punching.

Sturdily made from the finest materials, the Car Ramp/Magazine Binder offers you a lifetime or enjoyable use, unless of course you actually try to drive your car up onto them. They are available for $9.25 each — don't forget to specify which one you would like — and Ontario residents, must add 7% Provincial Sales Tax. Order from:

Computing Now! Binders
Suite 601
25 Overlea Blvd.,
Toronto, Ontario
M4H 1B1

See Order Form in this Issue.

SPECIAL PUBLICATION

Your First Computer

100 Pages **$3.95**

Your First Computer is a special guide from Computing Now! magazine for people who would like to get into computers but don't know where to begin. The publication includes articles on buying your first computer, buying a computer secondhand, a section on computer languages for beginners, surveys of computers and printers under $1000.00, and a complete glossary of computer terms.

ORDER TODAY FROM
Moorshead Publications, 25 Overlea Boulevard, Toronto, Ont., M4H 1B1 (416) 423-3262
Postage and Handling included.
Credit Card payment accepted

Keyboard Design

Before you can process those words, zap those aliens, spread those sheets or file away your list of exotic lizard skin cowboy boot manufacturers for future reference you have to have a keyboard. The complexities of this most fundamental peripheral are scrutinized in this article.

by Rani Lueder and M. Franz Schneider

There a number of ways of communicating with computers. Joysticks, mice, light pens and touch panels are a few. Optical scanners, machines that read the information on printed pages directly by interpreting patterns of light and dark, are beginning to reduce the demand for keyboard entry of text and data. Finally, we are approaching the day that we will be able to talk to computers and have them talk back to us. However, right now almost everybody interacts with micros through keyboards. As microcomputer technology continues to expand its scope and power, it seems reasonable to expect that we will begin to interact with them on an almost daily basis.

As the keyboard is anything but a trivial aspect of the lifestyle of someone who makes intensive use of a micro, its design and human engineering is, or at least should be, of considerable concern.

Horseless Carriages

The evolution of technology always carries over some of its previous phases. This is called stereotype in design. An example of the phenomenon is the initial placement of engines in automobiles. It seems peculiar that the engine is in the front of the car when the wheels it drives are generally in the back. However, the engine was intuitively placed at the front of the car that was where the earlier form of motive power, the horse, was generally found.

The keyboard of a computer imitates that of a typewriter for much the same reason. However, the structure of the typewriter keyboard is a function of the necessary placement of myriad gears and levers to make the mechanical workings of the machine operate. Computer keyboards, which are wholly electrically operated devices, are not bound by these constraints. They don't have to look like typewriter keyboards and, what's more, probably shouldn't.

For example, it is unnecessary, and even detrimental, to type on a single keyboard. The distance between the elbows, when they're hanging comfortably, and the keyboard is substantial, particularly for men. This can cause a great deal of discomfort as the forearm is forced to bend inwards to reach the keys.

In addition, the placement of the keys forces the hands to bend outwards in relation to the arms. A much better design would provide two separate keyboards, one for each hand. Minimally, the design should conform to a V shape to allow a more natural work posture.

Keyboard heights can also exact physical tolls. Operating a keyboard that is too high causes shooting pains in the arms and shoulders in only a short time. This is more pronounced in better typists, as these operators tend to maintain their arms in elevated and unsupported positions for prolonged periods of time. It can also cause muscle degeneration and arthritis over the long term.

If the keyboard is too low there is not enough knee room for tall operators. This results in strange contortions of posture over the working day. Research also indicates that performance is affected. Unless the

Keyboard Design

desk which holds the terminal one sits at all day is adjustable, it will fail to accommodate different sized users. A support surface adjustment range of twenty–four to thirty inches is probably sufficient. A range up to thirty–two inches would also provide for users who are restricted to wheelchairs.

It is not, however, feasible to provide an adjustable terminal for everyone. As a result, some standards recommend extremely thin keyboards to both increase the knee clearance for the tall operators and to reduce the working level height for short ones. To accomplish this, it is necessary to reduce keyboard angles, and in fact, a number of recommendations suggest extremely low keyboard angles.

This, as it turns out, has some undesireable consequences. An overview of the last twenty years of research shows conclusively that although users differ substantially in their preferred keyboard angles, the low angles of five degrees or less are liked the least. A recent study by IBM even indicates that performance is improved somewhat by greater keyboard angles. Eighteen debgress was most preferred and equal in performance benefits with twelve.

The best angle for the keyboard seems to be a function of the size of the operator's hands. People with smaller hands prefer greater keyboard angles to reduce the travel distance for their shorter fingers to reach the keys. Likewise, people with larger hands prefer lower keyboard angles to better accommodate the finger to key travel distance. How much of a difference keyboard angle adjustment will make to your productivity is not known.

It may not always be cost effective to buy a fully adjustable work station. On the other hand, many terminal manufacturers now offer adjustable keyboards that are worth considering.

Hand In Hand

Palm supports may be useful for reducing hand, arm, and shoulder strains if the operator intermittently uses them as rests. However, others find that they interfere with work. The best design would allow the supports to be moved out of the way. Ideally, they would also adjust in length to accommodate the lengths of assorted sizes of hands.

The importance of the keyboard as a visual element is of concern because frequent up and down head movements cause excessive stress to the spine. Some authorities consider it to be the primary visual source, as it is looked at more than half the time. Others maintain that it has much less influence on the work process, since these appear to be quick glances out of one's peripheral vision to make sure that the hands are positioned correctly. It seems to be looked at more by certain operators, such as novice typists and those involved in alphanumeric entry.

With so many new function keys, expert typists are also beginning to glance down more frequently. Characteristics of the keyboard can influence how easily the information is located and interpreted.

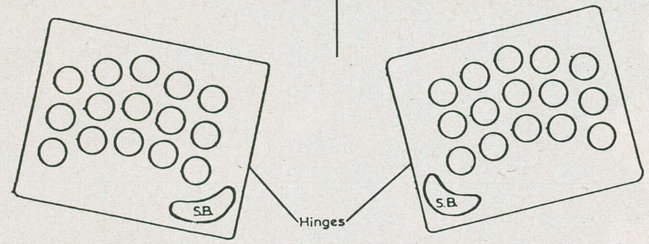


Figure 1. Separated and angled keyboard halves improve work postures.

Considerable thought should be given to the color and reflectiveness of the keyboard. Continuously alternating between a dark keyboard and the typically white paper one is typing from is uncomfortable because the eye must continually readjust itself to the two widely differing luminances. Gray keyboards with a matt finish are much less distracting. Color coding can also help provide meaning and structure to the keys.

Keyboard layout is another important factor in the ease of use inherent in a keyboard. The standard format we see on typewriters and keyboards, often called QWERTY after the first six letters in the first row of alphabetic keys, was developed by Sholes in 1878 to avoid jamming up the typewriter. It was not conceived primarily for operator convenience.

The ideal keyboard layout would locate the primary keys in the center row, where fingers naturally rest, and would either distribute the burden between the left and the right hand or place a slightly greater emphasis on the right side. In contrast, the Sholes layout overloads the left hand and places too much emphasis on the back row. The hand is often forced to skip between the front and back rows.

A number of alternate layouts have been proposed but the acceptance of any of these has been slow because of the costs involved in retraining typists already proficient on the Sholes keyboard. However, one format devised by Dvorak in 1936 is gaining acceptance and has recently been approved by ANSI, the American National Standards Institute, as an alternative to the conventional Sholes design.

The Dvorak keyboard places seventy percent of the most frequently used characters in the middle row and has a slight emphasis on the right side. Comparisons between it and the Sholes keyboard have not been consistent, however, one study found that the Dvorak style increased productivity by seventy when inexperienced or specially trained operators were used. However, it takes about twenty–eight days for a Sholes typist to reach the same keying rate on the Dvorak.

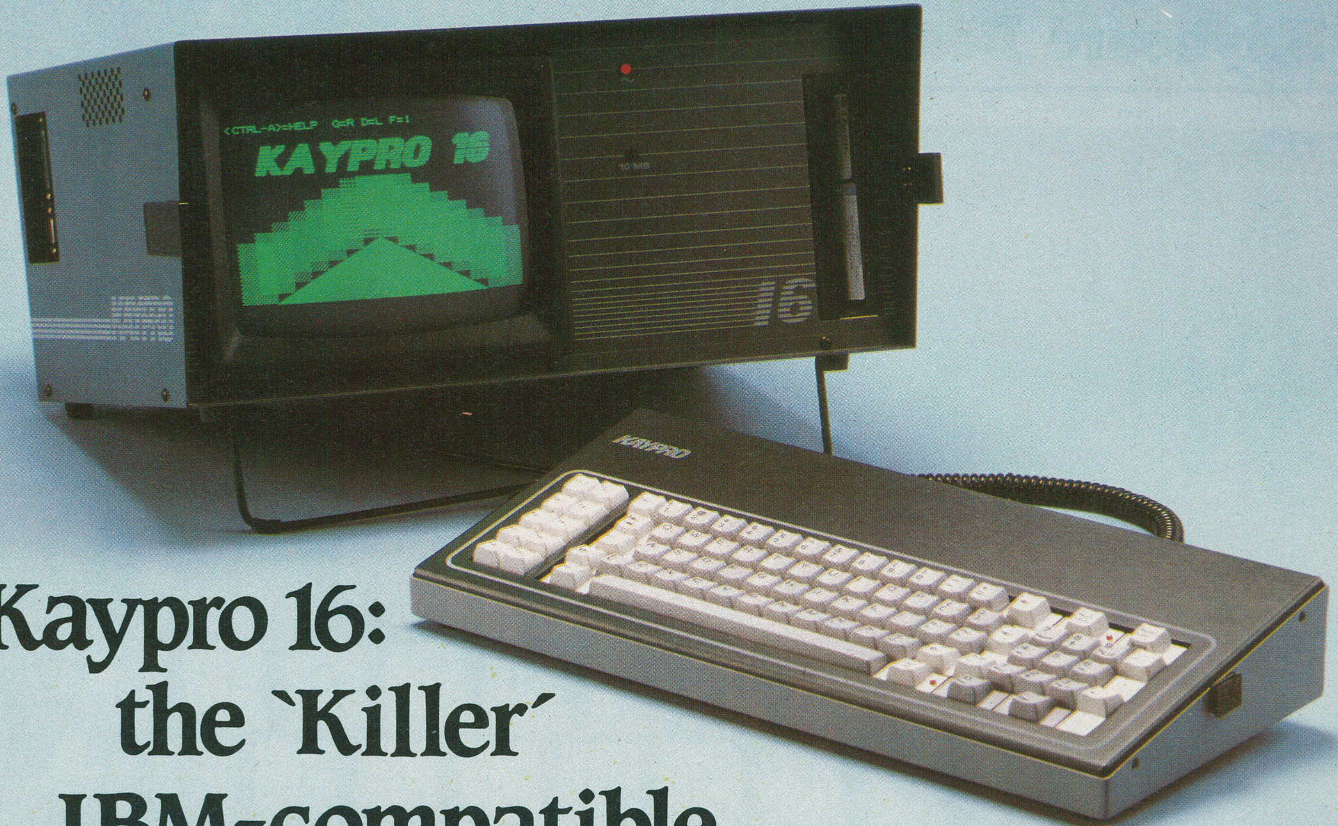
Numbers

The optimum numeric layout for a keyboard depends upon the task to which it is to be applied. For standard text entry the numbers typically on the top line will serve for the occasional digits which need be keyed. In fact, very few typists will use a numeric keypad in these instances, as the keys are typically too far to one side. However, if data entry is involved it is best to provide an additional numeric keypad.

If you compare the placement of the numbers of a calculator keypad with that of a touch telephone, you will notice that the calculator provides the small numbers at the bottom while the telephone places the small numbers at the top. The zero is always on the very bottom. This was a mistake on the part of the calculator manufacturers who had little awareness of human factors at the time that this style was implemented.

Although the telephone configuration has been shown to be more efficient, for both increasing speed and reducing errors, ultimately, the optimum format depends on the task at hand. If the application in question involves reference to a calculator, using a computer keypad which differs markedly from this will lead primarily to confusion. Likewise, if your work involves frequent telephone calls while operating a terminal, you should try to keep this format consistent.

There are other inconsistancies in traditional keyboard design. The space bar takes up far too much space considering that it is



Kaypro 16: the 'Killer' IBM-compatible beats the competition

At a time when technical innovation is so abundant it sometimes causes a yawn, Computron is proud to introduce the Kaypro 16 to Canada.

Kaypro 16 has features even the most blasé will sit up and notice. The Kaypro 16 is totally IBM-compatible. It actually enjoys the so-called "Killer" software others choke on. And, the Kaypro 16 can safely boast the lowest price of all competitive systems on the Canadian market today.

If that's not enough, count the standard features:

- 10 Mb hard disk plus 360K floppy
- 256K Bytes, expandable to 640K Bytes without further memory board purchases
- Serial and parallel ports
- Built-in monochrome graphics screen with RGB and composite video interfaces built-in for use with an external colour monitor
- Totally IBM-compatible keyboard
- Comprehensive name-brand software including MS-DOS 2.11, GW-Basic™, WordStar™, MailMerge™, CalcStar™, InfoStar +™ (DataStar™, ReportStar, FormSort), StarBurst™ and MITE™ Communications – all with full interactive tutorials.

Now, you ask, what does all this cost? Just $5195 or less.

Call or write today for a dealer near you and see for yourself why Kaypro 16 is a "Killer."

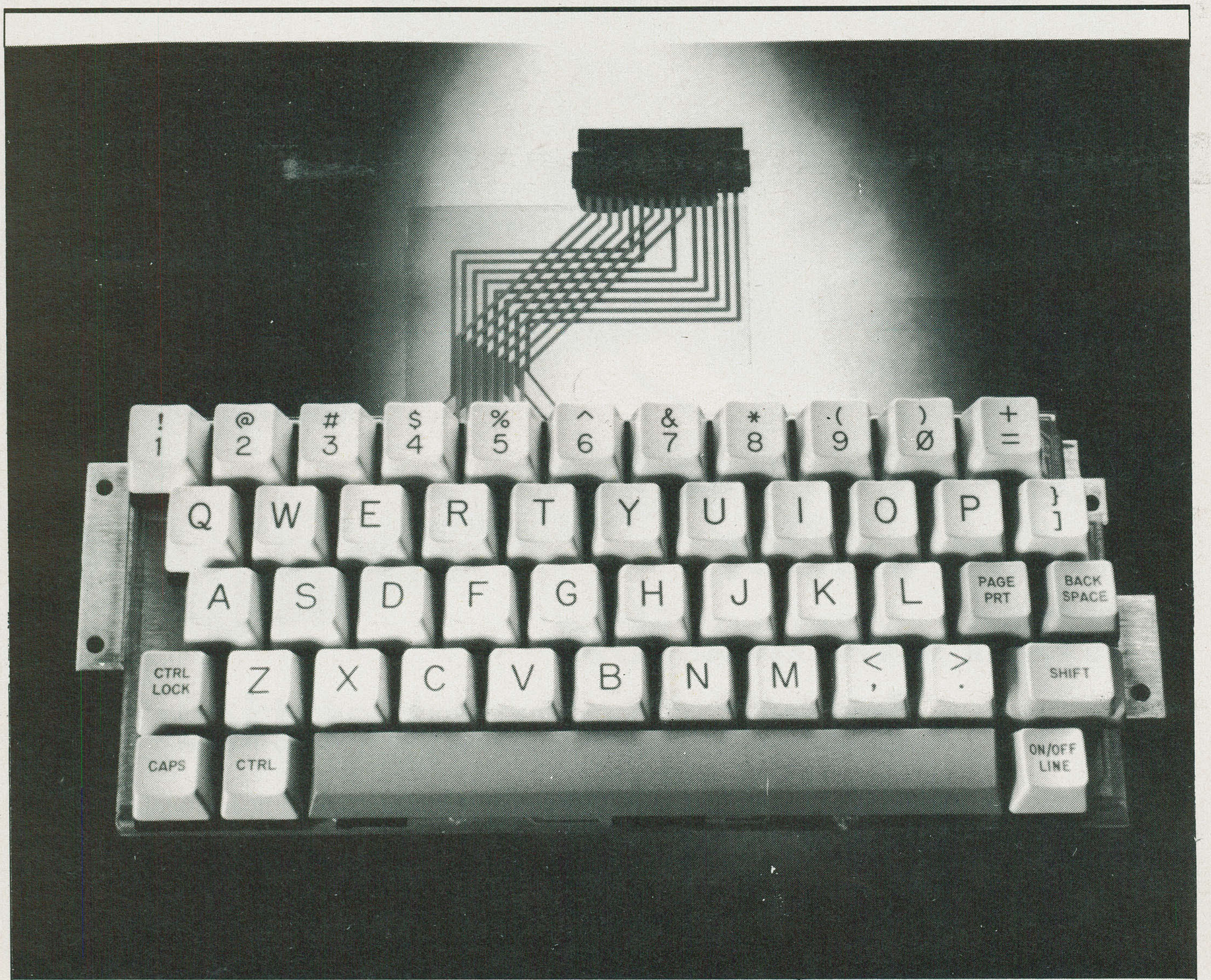
TORONTO: 55 Torbay Road
Unit #2, Markham, Ontario L3R 1G7
(416) 477-0828

HEAD OFFICE: 17507-107 Avenue
Edmonton, Alberta T5S 1E5
(403) 489-8400 Telex: 037-42881

Circle No. 9 on Reader Service Card

Computron

Keyboard Design



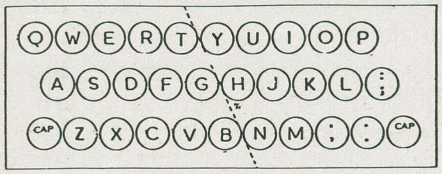


Figure 2. The standard Sholes (QWERTY) layout was designed to prevent jamming of the keys in the earliest typewriters.

usually only pressed on a small area under the right thumb. The carriage return is located more remotely than its function deserves. More specialized keyboards usually introduce more esoteric peculiarities, such as in the case of the IBM PC keyboard, which has its shift key in the wrong row.

Extra programmable keys are good for reducing the number of keystrokes required of an operator if the task to which the computer will be placed is highly repetitive and dictates frequent use of certain strings of numbers or letters. However, too many function keys, or too many permutations of shifts and supershifts in conjunction with the function keys, will serve largely to confuse the operator.

There are recommended values for the dimensions and distances between the keys, but you probably don't need to bother with them. Just put your hands on the keys... or have your secretary do it if she'll be using the machine primarily... and try them out.

Finally, make sure that your keyboard provides good feedback. Unless you are an infrequent typist, forget those touch sensitive keyboards... the feedback is not good enough. It should be both auditory, usually a click which simulates a mechanical keystroke, and tactile, such that it snaps when a key has been depressed sufficiently. Research shows that feedback can help both speed and error reduction. If you doubt this statement, try kissing somebody without smacking at the end...it just isn't the same. **CN!**

Product Mart

Where Buyers Find Sellers

"PROJECT GENESIS" (64K CoCo) $24.95. Now shipping! UPS, cables, software, ribbon re-inkers. Apple IIc or CoCo parallel printer interface $99.95. Catalog $1.00. **BETTER SOFTWARE,** Dept. ETI (or CN!) Box 2203, Charlottetown, P.E.I. C1A 8B9.

APPLE II Disk owners to trade programs. Please send list and/or disk(s) to **WALTER McWILLIAMS,** 345 Country Club Dr. Belleville, Ontario, Canada K8R 1B5.

OVER 100 kits. The best books on the market. Satisfaction guaranteed. For free catalogue, send self addressed stamped envelope with 64¢ postage to **SIMPLIFIED TECHNOLOGIES,** 214 King St., East, Suite 301, Hamilton, Ont., L8N 1B5.

VIDEOX — VIDEX 80 col. lookalike board. No annoying video flicker during scrolling. Almost zero power when idle. 560 x 200 hires option. PCB $39.95. Videx to Videox upgrade kit $89.95. Complete kit $139.95 assembled and tested $159.95. All with documentation plus 7% tax in Ontario. Money Order or Certified Cheque. **MITINA ENTERPRISES,** P.O. Box 245, Woodbridge, Ontario. L4L 1B1.

"PUBLIC domain software for APPLE II+, IBM-PC, for sale at $10.00/disk. Please specify computer. **J.R. Hebert,** P.O. Box 663, Sutton, P.Q. J0E 2K0."

ONE 3 years old Kennedy 9832, 800 BPI 9 Track Data Recorder for sale. For details contact **BRUCE McCORMACK,** Brock University, St. Catharines, Ontario. 416-688-5550

HAYES

Smartmodem 300	**$299.00**
Smartmodem 1200	**$699.00**
Smartmodem 1200B	**$599.00**
Micromodem IIe	**$359.00**
Chornograph	**$293.00**
ANCHOR Mark VII	**$179.00**
Mark XII 1200 BPS	**$399.00**
Commodore 1650 Direct	**$169.00**
Atari 1030 Direct	**$99.00**

APPLE DRIVES

RANA Elite I	**$339.00**
RANA Elite II, 360K	**$479.00**
RANA Elite III, Quad Density	**$679.00**
Micro SCI A-2	**$282.00**

IBM DRIVES

Shugart SA 455-3 1/2ht, 360K	**$225.00**
Panasonic 1/2ht. 360K	**$209.00**
TEAC 1/2ht, 360K	**$219.00**
TandonTM 100-2, 360K	**$259.00**
MPI B-52, 360K special	**$179.00**

EPSON

RX-80	**$376.00**
RX 80/FT	**$459.00**
RX 100	**$676.00**
FX 80 w/tractor	**$645.00**
JX 80 Color	**$899.00**
LQ 1500	**$1699.00**
Panasonic 1090/1091	**$339.00/439.00**
Okidata 92/93	**$599.00/969.00**
Brother HR15/HR25	**$599.00/995.00**
Silver Reed 400/550	**$449.00/599.00**

MAXELL

MD-1 SS/DD . **$29.95.** MD-2 DS/DD . **$39.95.**
64K IBM PC/XT upgrade set **$67.99**

640K IBM PC/XT upgrade board .... $395, Upgrade your PC/XT to its full 640K potential with 640K board. A must for all PC/XT owners at this reduced price! Reg. **$199** now 1/2 price!

Dealer Inquiries Invited
Add 4% Shipping & Handling
$3 on disks

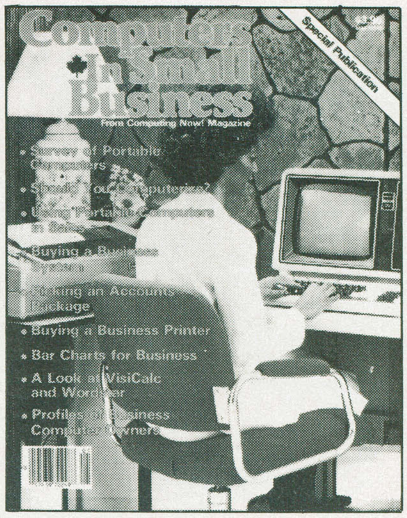
MICRO NET WORK, Route 1, Anola MB R0E 0A0

Advertising in the Product Mart section allows you to reach thousands of Canadian readers nation wide effectively and economically. Please contact Dierdre Whitehead, Advertising Service Manager, (416) 423-3262, for further details on the various sizes and styles available.

WHAT DO YOU DO?
Send us your typewritten or clearly printed words, your permanent address and telephone number and your payment (no cash please). Make your cheque or money order payable to Moorshead Publications. We're at Suite 601, 25 Overlea Blvd., Toronto, Ontario, M4H 1B1.

WHAT DO WE DO?
We typeset your words (and put the first word and your company name in **BOLD** capital letters). Your advertisement will appear in the first available issue.

Computers In Small Business



Small companies across Canada are hungry for knowledge about the microcomputer revolution. Many of them believe that they alone are ignorant about what is going on and what equipment is available.

This Special publication is addressed exclusively to this market. The articles comprise reprints of the very best material already published in *Computing Now!* magazine together with several specially commissioned features to form a well balanced publication. We believe this Special is of real use to the hundreds of thousands of small companies on the verge of buying a microcomputer.

First Published March 1984

$4.00 plus $1.00 postage and handling For a copy call (416) 423-3262 or write:

**Moorshead Publications
25 Overlea Boulevard,
Suite 601,
Toronto, Ontaro, M4H 1B1**

COMPUTER PRESS

SuperScript is a word processor enhancement program which lets users create complex mathematical expressions using subscripts, superscripts, integrals, fractions and a variety of other special symbols and notations, including Greek letters. Developed by *Technical Support Software*, the program works with most PC compatible word processing programs...
Circle No. 51 on Reader Service Card.

The **Linkup** family of products comprises of an intelligent plug-in communications board and related software to connect IBM PCs, XTs and compatibles to a variety of mainframe computers or other PCs. *Information Technologies*' communications boards are supplied with a wide variety of asynchronous and synchronous protocols...

Circle No. 50 on Reader Service Card.

Formic Videotex Systems of Montreal has announced a series of NAPLPS (Telidon) software decoders for the Apple //e and //c. The **SOFDEC** series was created to establish communications via modem with NAPLPS videotex databases. SOFDEC 'C', for the //c, is a software product, and SOFDEC 'E', for the //e, consists of firmware on an RS-232C serial card...

Circle No. 48 on Reader Service Card.

Apricorn, of San Diego, California has announced two new peripheral cards for the Apple // series computers. The **Extend-it** is a 64K memory module which can be added to an 80-column text card equipped Apple //e. The **Super Serial Imager** has graphics dump capabilities that enable Apple // computers to transfer high resolution images from the screen to a dot-matrix printer. In addition, the card also has built-in communications firmware and supports a variety of 300/1200 baud intelligent modems such as models from Hayes, Novation and Anchor Automation...

Circle No. 47 on Reader Service Card.

The **PC*VISION*** Frame Grabber from *Imaging Technology Incorporated* is a board level digitizer and display module which is plug-compatible with the IBM Personal Computer. The module captures an RS-170 video signal (TV standard) at a rate of 30 frames per second, stores the image in an on-board 512 by 512 by 8-bit frame memory, and displays the stored image on an external monitor...
Circle No. 46 on Reader Service Card.

Addresses: TRW Data Systems, Division of TRW Canada Limited, 270 Yorkland Boulevard, Willowdale, Ontario M2R 1R8 (416) 491-9606 ● Formic Videeotex Systems, 8571, St. Denis, Montreal, Quebec H2P 2H4 (514) 384-2655 ● Imaging Technology, 600 West Cummings Park, Woburn, MA 01801 (617) 938-8444 ● cLINE, 20 West Chicago, IL 60610 (312) 944-4510 ● Epson Canada 285 Yorkland Boulevard, Willowdale, Ontario M2J 1S5 (416) 495-9955 ● Batteries Included, 30 Mural Street, Richmond Hill, Ontario L4B 1B5 ● Technical Support Software, P.O. Box 289, Chestnut Hill, MA 02167 (617) 734-4130 ● Microcom, 1400A Providence Highway, Norwood, MA 02062 (617) 762-9310 ● Apricorn, 7050 Convoy Court, San Diego, CA 92111 (619) 569-9483 ● Harvard Associates, Inc., 260 Beacon Street, Somerville, Massachusetts 02143 (617) 492-2999 ● Shugart Corporation, 475 Oakmead Parkway, Sunnyvale, California 94086 (Markham, Ontario: 416-475-2655) ● Systems Manufacturing Technology, Incorporated (SMT), 1145 Linda Vista Drive, #109, San Marcos, California 92069 (619) 744-3590 ● Information Technologies Incorporated, 7850 East Evans Road, Scottsdale, Arizona 85260 (602) 998-1033 ● CH Products, 1558 Osage Street, San Marcos, California 92069 (619) 744-8546 ● Catspaw, Incorporated, P.O. Box 1123, Salida, Colorado 81201 (303) 539-3884

Advertisers Index

The last thing your computer will see before it goes crazy

You or someone else charged with static electricity can drive your personal computer crazy. Garbled date, memory loss, video interference and mechanical malfunctions are among the symptoms of a PC that's "a little touched."

And a little touch is all it takes. Your PC is more sensitive than you when it comes to static. A charge in your finger so small that you can't see or feel it can do the damage.

Static is ever-present. Even when the humidity is up, you can generate a charge by walking on a carpet or vinyl floor. The resulting glitches or downtime could be enough to make you even a little mad.

Now, a First Touch™ Static Control Computer Pad can keep your PC sane. The pad drains away static charge with your first and every touch. The specially formulated 3-layer construction also cushions the equipment, prevents slippage and reduces noise.

The First Touch pad is an evolution of a 3M technology created for the microelectronics industry. Chances are, the circuits in your computer were protected during manufacture and shipments by 3M static control products.

Proven effective in industry, easily installed by you in home or office, The First Touch pad is available in 24" x 26" and 9½" x 23½." . Look for our packages.

The first thing to do to keep it sane

... For the name of your nearest dealer contact:

Static Control Systems,
Post Office Box 5757
London, Ontario
N6A 4T1
or call
(519) 452-6138

First Touch™
Static Control Computer Pad

3M Hears You...

3M

Circle No. 12 on Reader Service Card

84-12-EH-0224-E

You're traveling through 140° terrain at 300 rpm.

Only one disk guarantees safe passage through the torrid zone of drive heat. Maxell.

A lifetime warranty. And manufacturing standards that make it almost unnecessary.

Consider this: Every time you take your disk for a little spin, you expose it to drive heat that can sidetrack data. Worse, take it to the point of no return. Maxell's unique jacket construction defies heat of 140°F. And keeps your information on track.

And Maxell runs clean. A unique process impregnates lubricants throughout the oxide layer. Extending media and head life. How good is Gold?

Maxell's the disk that many drive manufacturers trust to put new equipment through its paces. It's that bug-free.

So you can drive a bargain. But in accelerated tests, Maxell floppys lead the industry in error-free performance and durability. Proving that if you can't stand the heat you don't stand a chance.

maxell®

IT'S WORTH IT.

Griffco

Griffco Marketing Inc.

Griffco Marketing Inc., P.O. Box 1218, Station B, Mississauga, Ontario L4Y 3W5 (416) 625-6559

Circle No. 24 on Reader Service Card